ATMIC NETWORKS
YOUR GUIDE TO GLOBAL CERTIFICATION

"Please note that these files may not be up to date. However, the questions will help you understand the exam format and typical question patterns."

www.atmicnetworks .com

Warning: Keep connected with our support team for latest updates

Question: 1

What are the four fundamental elements that make a language?

A. An alphabet, phonetics, phonology, and semantics
B. An alphabet, a lexis, phonetics, and semantics
C. An alphabet, morphology, phonetics, and semantics
D. An alphabet, a lexis, a syntax, and semantics

Answer: D

Explanation:

Topics: language alphabet lexis syntax semantics
Explanation:
We can say that each language (machine or natural, it doesn't matter) consists of the following elements:
An alphabet:
a set of symbols used to build words of a certain language
(e.g., the Latin alphabet for English,
the Cyrillic alphabet for Russian, Kanji for Japanese, and so on)
A lexis:
(aka a dictionary) a set of words the language offers its users
(e.g., the word "computer" comes from the English language dictionary, while "cmoptrue" doesn't;
the word "chat" is present both in English and French dictionaries,
but their meanings are different)
A syntax:
a set of rules (formal or informal, written or felt intuitively)
used to determine if a certain string of words forms a valid sentence
(e.g., "I am a python" is a syntactically correct phrase, while "I a python am" isn't)
Semantics:
a set of rules determining if a certain phrase makes sense
(e.g., "I ate a doughnut" makes sense, but "A doughnut ate me" doesn't)

Question: 2

What will be the output of the following code snippet?

```
x = 1
y = 2
z = x x = y y = z print(x, y)
```

A. 1 2
B. 2 1
C. 1 1
D. 2 2

Answer: B

Explanation:

Topic: copying an immutable object by assigning Try it yourself:

```
x = 1
y = 2
z = x print(z) #    1
x = y print(x) #2
y = z print(y) #1
print(x, y)   #2  1
```

Explanation:
Integer is an immutable data type.
The values get copied from one variable to another.
In the end x and y changed their values.

Question: 3

Python is an example of:

A. a machine language
B. a high-level programminglanguage
C. a natural language

Answer: B

Explanation:
Topic: high-level programming language
Explanation:
https://en.wikipedia.org/wiki/Python_(programming_language)

Question: 4

What will be the output of the following code snippet? `print(3 /  5)`

A. 6/10
B. 0.6
C. 0
D. None of the above.

Answer: B

Explanation:
Topic: division operator
Try it yourself: `print(3 / 5)  # 0.6`

```
print(4 / 2)  # 2.0
```

Explanation:
The division operator does its normal job.
And remember the division operator ALWAYS returns a float.

Question: 5

Strings in Python are delimited with:

A. backslashes (i.e., \)
B. double quotes (i.e., ") or single quotes (i.e., ')
C. asterisks (i.e., *)
D. dollar symbol (i.e., $)

Answer: B

Explanation:
Topics: strings quotes
Try it yourself:

```
print("Hello")  # Hello
print('World')  # World
```

Explanation:
Unlike in other programming languages, in Python
double quotes and single quotes are synonyms for each other.
You can use either one or the other.
The result is the same.

Question: 6

What will happen when you attempt to run the following code? `print(Hello, World!)`

A. The code willraise theSyntaxError exception.
B. The code willraise theTypeError exception.
C. The code willraise theValueError exception.
D. The code willprint Hello, World! to the console.
E. The code willraise theAttributeError exception.

Answer: A

Explanation:
Topics: print() SyntaxError
Try it yourself:

```
# print(Hello, World!)
# SyntaxError: invalid syntax
```

Explanation:
The exclamation mark makes it a syntax error.

Question: 7

A function definition starts with the keyword:
A. def
B. function
C. fun

Answer: A

Explanation:
Topic: def
Try it yourself:

```
def my_first_function():
print('Hello')
my_first_function() # Hello
```

Explanation:
https://www.w3schools.com/python/python_functions.asp

Question: 8

Assuming that the tuple is a correctly created tuple, the fact that tuples are immutable means that the following instruction: `my_tuple[1] = my_tuple[1] + my_tuple[0]`

A. can be executed if and only if the tuple contains at least two elements
B. is illegal
C. may be illegal if the tuple contains strings
D. is fully correct

Answer: B

Explanation:

Topics: dictionary

Try it yourself:

```
my_tuple =  (1, 2, 3)
my_tuple[1] = my_tuple[1] + my_tuple[0]
# TypeError: 'tuple' object does not support item assignment
```

Explanation:

A tuple is immutable and therefore you cannot assign a new value to one of its indexes.

Question: 9

What is the expected output of the following code?

```
def func(x):
return 1 if x % 2 != 0 else 2
print(func(func(1)))
```

A. The code is erroneous.
B. None
C. 2
D. 1

Answer: D

Explanation:

Topics: def conditional expression (if else) modulus operator not equal to operator

Try it yourself:

```
def func(x):
return 1 if x % 2 != 0 else 2
print(func(func(1)))   # 1
print(1 % 2)           # 1
print(1 % 2 != 0)      #True
```

Explanation:

This is a conditional expression.
1 % 2 is 1 and therefore not equal to 0
The condition is True and the inner func() function call returns 1
That 1 is passed to the outer function which will also return 1

Question: 10

Take a look at the snippet, and choose the true statements:

```
nums =  [1, 2,  3]
vals = nums del vals[1:2]
```

(Select two answers)

A. nums is longer than vals
B. nums and vals refer to the same list
C. vals is longer than nums
D. nums and vals are of the same length

Answer: B,D

Explanation:

Topics: list referencing list slicing del

Try it yourself:

```
nums = [1, 2, 3]
vals = nums del vals[1:2] print(nums)   # [1, 3]
print(vals)  # [1, 3]
```

Explanation:

A list is a mutable data type.

Assigning a mutable data type creates a reference to the same object. vals and nums will point to the same object in the memory and when you change one you automatically change the other, too.

Question: 11

What is the output of the following code?

```
a = 1
b = 0
x = aor b
y = not(a and b)
print(x + y)
```

A. The program will cause an error B. 1

C. The output cannot be predicted D. 2

Answer: D

Explanation:

Topics: logical operators booleans addition operator implicit type casting

Try it yourself:

```
a = 1
b = 0
x = a or b print(x)#       1
print(1 or 0)    #  1
y = not(a and b) print(y)  #      True
print(1 and 0)   #  0
print(not 0)     #  True
print(x + y)     #  2
print(1 + True)  #  2
```

Explanation:

If you calculate with a boolean True becomes the integer 1
and therefore 1 + True is 2

Question: 12

What is the output of the following snippet?

```
dct = {}
dct['1'] = (1, 2)
```

```
dct['2'] = (2, 1)
for x in dct.keys(): print(dct[x][1], end='')
```

A. 21
B. (2,1)
C. (1,2)
D. 12

Answer: A

Explanation:

Topics: dictionary tuple indexing for dict.keys() print()
Try it yourself:

```
dct = {}
dct['1'] = (1, 2)
dct['2'] = (2, 1)
print(dct) #  {'1': (1, 2), '2': (2, 1)}
for x in dct.keys():
print(dct[x][1], end='')  # 21
print()
print(dct['1'][1])  # 2
print(dct['2'][1])  # 1
```

Explanation:

```
dct.keys() are  the keys'1' and '2'
dct['1'][1] is  2 and
dct['2'][1] is  1
```

Question: 13

What would you insert instead of ???
so that the program checks for even numbers?

```
if ???:
print('x is an even number')
```

A. x % x == 0
B. x % 1 == 2
C. x % 2 == 0
D. x % 'even' == True
E. x % 2 == 1

Answer: C

Explanation:

Topics: if modulus operator equal to operator
Try it yourself:

```
x = 4
if x % 2 == 0:
print('x is an even number') # x is an even number
```

Explanation:
Every number that divided by two does not leave a rest is even.

Question: 14

What is the expected output of the following code?

```
x = [0, 1, 2]
x.insert(0, 1) del x[1] print(sum(x))
```

A. 3
B. 2
C. 4
D. 5

Answer: C

Explanation:

Topics: insert() del sum() list Try it yourself:

```
x = [0, 1, 2]
x.insert(0, 1)
print(x) del   # [1, 0, 1, 2]
x[1] print(x)
print(sum(x))  # [1, 1, 2]
               # 4
```

Explanation:

list.insert(i, x)

insert() inserts an item at a given position.

The first argument is the index of the element before which to insert. insert(0, 1) inserts 1 before index 0 (at the front of the list).

The del keyword deletes the given object.

In this case x[1]

The sum() function adds the items of a list (or a different iterable) and returns the sum.

Question: 15

The digraph written as #! is used to:

A. tell a Unix or Unix-like OS how to execute the contents of a Python file.
B. create a docstring.
C. make a particular module entity a private one.
D. tell an MS Windows OS how to execute the contents of a Python file.

Answer: A

Explanation:

Topics: #! shebang

Explanation:

This is a general UNIX topic.

Best read about it here:

https://en.wikipedia.org/wiki/Shebang_(Unix)

Question: 16

What is the expected output of the following code?

```
data = ['Peter', 404, 3.03, 'Wellert', 33.3]
print(data[1:3])
```

A. None of the above.
B. ['Peter', 404, 3.03, 'Wellert', 33.3]
C. ['Peter', 'Wellert']
D. [404, 3.03]

Answer: D

Explanation:

Topic: list indexing

Try it yourself:

```
data = ['Peter', 404, 3.03, 'Wellert', 33.3]
print(data[1:3]) #  [404, 3.03]
```

Explanation:
You have a list of five elements of various data types.
[1:3] slices inclusive the first index and exclusive the third index.
Meaning it slices the first and second index.

Question: 17

What will be the output of the following code snippet?

```
d = {}
d[1] = 1
d['1']  =2
d[1] += 1
sum = 0
for k  ind:
sum += d[k] print(sum)
```

A. 3
B. 4
C. 1
D. 2

Answer: B

Explanation:
Topics: for dictionary indexes add and assign operator Try it yourself:

```
d = {}
print(d) #  {}
d[1] = 1
print(d)   #  {1:   1}
d['1'] = 2
print(d)   #  {1:   1, '1':      2}
d[1] += 1
print(d)   #  {1:   2, '1':      2}
sum = 0
for k ind:
sum += d[k]
print("key: ", k, " - value: ", d[k])
#  key: 1   - value: 2
print(sum) #  4
sum = 0
for k in d.keys(): sum += d[k]
print("key: ", k, " - value: ", d[k])
#  key: 1   - value: 2
print(sum) #  4
```

Explanation:
The knowledge you need here is that a dictionary
can have indexes of different data types. Therefore d[1] is a different index than d['1'] and they can both exist in the same dictionary. To iterate through a dictionary is the same as iterating through dict.keys()
In k will be the keys of the dictionary.
In this case 1 and '1'
The value of the first key will be 2
and the value of the other key will also be 2 and therefore (the) sum is 4

Question: 18

How many stars will the following snippet print to the monitor?

```
i = 4
while i > 0:
i -= 2
print('*') if i == 2:
break
else: print('*')
```

The snippet will enter an infinite loop.

A. 0
B. 2
C. 1

Answer: C

Explanation:

Topics: if while break else (nobreak)

Try it yourself:

```
i = 4
while i > 0:   # i is 4
i -= 2      # i is 2
print('*') # *
if i == 2: # Yip, i is 2
break # Leave the loop directly
else:          # Does not apply, because the break got triggered
print('*')
```

Explanation:

In the first iteration the break gets directly triggered. Therefore there will be only one star.

The else would only apply, if the break does NOT get triggered.

Question: 19

What is CPython?

A. It' a programming language that is a superset of the C language,

B. designed to produce Python-like performance with code written in C
C. It' a programming language that is a superset of the Python,
D. designed to produce C-like performance with code written in Python
E. It's a default, reference implementation of the C language, written in Python
F. It's a default, reference implementation of the Python language, written in C

Answer: F

Explanation:

Topic: CPython

Explanation:

Guido van Rossum used the "C" programming language
to implement the very first version of his language and this decision is still in force.

All Pythons coming from the PSF (Python Software Foundation) are written in the "C" language.
There are many reasons for this approach.
One of them (probably the most important) is that thanks to it, Python may be easily ported and migrated to all platforms with the ability to compile and run "C" language programs (virtually all platforms have this feature, which opens up many expansion opportunities for Python).
This is why the PSF implementation is often referred to as CPython.
This is the most influential Python among all the Pythons in the world.
https://en.wikipedia.org/wiki/CPython

Question: 20

What is the expected output of the following code?

```
def func(p1, p2):
p1 = 1
p2[0] = 42
x = 3
y = [1, 2, 3]
func(x, y)
print(x, y[0])
```

A. 3 1
B. The code is erroneous.
C. 142
D. 342
E. 1 1

Answer: D

Explanation:
Topics: def list argument passing mutable vs. immutable
Try it yourself:

```
def func(p1, p2):
p1 = 1
p2[0] = 42
x = 3
y = [1, 2, 3]
func(x, y)
```

```
print(x, y[0])  #  3 42
```

Explanation:
This question is about argument passing.
It is a big difference, whether you pass a mutable or an immutable data type.
The immutable integer in x gets copied to p1
and the change of p1 does not effect x
The mutable list in y gets referenced to p2
and the change of p2 effect y

Question: 21

What is the expected output of the following code?

```
def func(data):
for d in data[::2]:
yield d
for x in func('abcdef'):
print(x, end='')
```

A. bdf
B. An empty line.
C. abcdef
D. ace

Answer: D

Explanation:
Topics: def yield for print() with end parameter list slicing
Try it yourself:

```
def func(data):
for d in data[::2]:
yield d
for x in func('abcdef'):
print(x, end='')  # ace
```

Explanation:
The generator function will return every second element of the passed data.

Question: 22

You develop a Python application for your company.
You have the following code.

```
def main(a, b, c, d): value = a + b * c - d return value
```

Which of the following expressions is equivalent to the expression in the function?

A. (a + b) * (c - d)
B. a + ((b * c) - d)
C. None of the above.
D. (a + (b * c)) - d

Answer: D

Explanation:
Topics: addition operator multiplication operator subtraction operator operator precedence

Try it yourself:

```
def main(a, b, c, d):
value = a + b * c - d       #   3
# value = (a + (b *  c)) - d #  3
# value = (a + b) * (c - d) # -3 # value = a + ((b    * c) - d) #   3
return value
print(main(1, 2,  3, 4)) #  3
```

Explanation:
This question is about operator precedence
The multiplication operator has the highest precedence and is therefore executed first.
That leaves the addition operator and the subtraction operator
They both are from the same group and therefore have the same precedence.
That group has a left-to-right associativity.
The addition operator is on the left and is therefore executed next.
And the last one to be executed is the subtraction operator

Question: 23

What is the expected behavior of the following program?

```
try:
print(5/0)
break
except:
print("Sorry, something went wrong...")
except (ValueError, ZeroDivisionError):
print("Too bad...")
```

A. The program will causea SyntaxError exception.
B. The program will causea ValueError exception and output the following message: Too bad...
C. The program will causea ValueError exception and output a default error message.
D. The program will raise an exception handled by the first except block.
E. The program will causea ZeroDivisionError exception andoutput a default error message.

Answer: A

Explanation:
Topics: try except break SyntaxError ValueError ZeroDivisionError
Try it yourself:

```
try:
print(5/0)
#  break
except:
print("Sorry, something went wrong...")
#  except (ValueError, ZeroDivisionError):
# print("Too bad...")
```

Explanation:
There are two syntax errors: break can not be used outside of a loop, and the default except must be last.

Question: 24

Which of the following variable names are illegal? (Select two answers)

A. TRUE
B. True
C. true
D. and

Answer: B,D

Explanation:

Topics: variable names keywords True and

Try it yourself:

```
TRUE =  23
true =  42
#  True = 7  # SyntaxError: cannot assign to True
#  and = 7   # SyntaxError: invalid syntax
```

Explanation:

You cannot use keywords as variable names.

Question: 25

What is the expected output of the following code?

```
z = y = x =  1
print(x, y, z, sep='*')
```

A. x*y*z
B. 111*
C. x y z
D. 1*1*1
E. 1 1 1
F. The code is erroneous.

Answer: D

Explanation:

Topic: multiple assignment print() with sep parameter

Try it yourself:

```
z = y = x =  1
print(x, y, z, sep='*')   # 1*1*1
print(x, y, z, sep=' ')   #  1 1 1
print(x, y, z)            #  1 1 1
```

Explanation:

The print() function has a sep parameter which stands for separator.

The default value of the sep parameter is a space character.

You can change it to anything you want.

Question: 26

What is the expected output of the following code?

```
def func(text, num):
while num >  0:
print(text) num = num -   1
func('Hello', 3)
```

A. An infinite loop.

B.

```
Hello
Hello
Hello
```

C.

```
Hello
Hello
Hello
Hello
```

D.

```
Hello
Hello
```

Answer: A

Explanation:

Topics: def while indentation

Try it yourself:

```
def func(text, num):
while num >  0:
print(text)
num = num -  1
func('Hello', 3)  # An infinite loop
```

Explanation:

The incrementation of num needs to be inside of the while loop. Otherwise the condition num > 0 will never be False

It should look like this:

```
def func(text, num):
while num >  0:
print(text)
num = num -  1
func('Hello', 3) """
Hello
Hello
Hello
```

Question: 27

What is the expected output of the following code?

```
x =True
y =False
z =False
if not x or y: print(1)
elif  not x or not y and z:
print(2)
elif  not x or y or not y and x:
print(3)
else:
print(4)
```

A. 4
B. 3
C. 2
D. 1

Answer: B

Explanation:

Topics: if elif else not and or operator precedence Try it yourself:

```
x = True y = False z = False # if not x or y: # if (not True) or False: # if False or
False: if False: print(1)
#  elif not x or not y and z:
#  elif (not True) or (not False) and False:
# elif False or True  andFalse
# elif False or (True and False
# elif False or False
elif False:
print(2)
#     elif not xor      y or not      y and x:
#  elif (not True) or False or (not False) and True:
#     elif Falseor False  or   True and True:
#     elif Falseor False  or (True  and True):
#     elif Falseor False  or     True:
#  elif (False or False) or True:
#     elif Falseor True:
elif True:
print(3) # 3
else:
print(4)
```

Explanation:
There are three operators at work here.
Of them the not operator has the highest precedence, followed by the and operator.
The or operator has the lowest precedence.

Question: 28

What is the expected output of the following code?

```
x =  [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]
def func(data): res = data[0][0] for da in data: for d in   da:
if res < d:
res = d return res
print(func(x[0]))
```

A. The code is erroneous.
B. 8
C. 6
D. 2
E. 4

Answer: E

Explanation:

Topics: def for if list
Try it yourself:

```
x =  [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]
def func(data):
print('data:', data)    #   [[1, 2], [3, 4]]
res = data[0][0]        #   1
print('res:', res) for da in data: print('da:', da)   #      [1,    2]     ->     [3,
```

```
                    4]
for d in da:
print('d:', d) #   1 -> 2 -> 3 -> 4
if res < d: res = d return res print(func(x[0]))      # 4
print(func([[1, 7], [3, 4]]))  # 7
```

Explanation:
This function looks for the highest element in a two dimensional list (or another iterable).
In the beginning the first number data[0][0] gets taken as possible result.
In the inner for loop every number is compared to the possible result.
If one number is higher it becomes the new possible result.
And in the end the result is the highest number.

Question: 29

Which of the following for loops would output the below number pattern?

```
11111
22222
33333
44444
55555
```

A.

```
for i in range(0, 5): print(str(i) *    5)
```

B.

```
for i in range(1, 6): print(str(i) *    5)
```

C.

```
for i in range(1, 6):
print(i, i, i, i, i)
```

D.

```
for i in range(1, 5): print(str(i) *    5)
```

Answer: B

Explanation:
Topics: for range() str() multiply operator string concatenation
Try it yourself:

```
for i in range(1, 6):
print(str(i) *  5)
```

```
11111
22222
33333
44444
55555
```

```
print('  ------  ')
for i in range(0, 5):
print(str(i) *  5)
```

```
00000
11111
```

```
22222
33333
44444

print('  ------ ')
for i in range(1, 6): print(i, i, i, i, i)
"""
1 1 1  1 1
2 2 2  2 2
3 3 3  3 3
4 4 4  4 4
5 5 5  5 5
print('  ------ ')
for i in range(1, 5):
print(str(i) *  5)
11111
22222
33333
44444
```

Explanation:
You need range(1, 6)
because the start value 1 is inclusive and the end value 6 is exclusive. To get the same numbers next to each other
(without a space between them) you need to make a string and then use the multiply operator string concatenation The standard separator of the print() function is one space. print(i, i, i, i, i) gives you one space between each number. It would work with print(i, i, i, i, i, sep='') but that answer is not offered here.

Question: 30

What is the output of the following snippet?

```
def fun(x, y, z):
return x +  2 * y + 3 * z
print(fun(0, z=1,  y=3))
```

A. the snippet is erroneous
B. 9
C. 0
D. 3

Answer: B

Explanation:
Topics: positional parameter keyword parameter operator precedence
Try it yourself:

```
def fun(x, y, z):
return x +  2 * y + 3 * z
print(fun(0, z=1,  y=3))        # 9
print(0 +  2 * 3 + 3  * 1)      #  9
print(0 +  (2 * 3) +  (3 * 1))  #  9
print(0 +  6 + (3  *1))         #  9
print(0  + 6 + 3)               # 9
print(6  + 3)                   # 9
print(9)                        # 9
```

Explanation:
The function here works fine.
The keyword arguments do not have to be in the correct order among themselves as long as they are all listed after all positional arguments.
And because multiplication precedes addition 9 gets returned and printed.

Question: 31

The value thirty point eleven times ten raised to the power of nine should be written as:

A. 30.11E9
B. 30E11.9
C. 30.11E9.0
D. 30.11*10^9

Answer: A

Explanation:
Topic: scientific notation
Try it yourself:

```
print(30.11E9)          #    30110000000.0
# print(30E11.9)        #   SyntaxError:   invalid      syntax
# print(30.11E9.0)      #   SyntaxError:   invalid      syntax
# print(30.11*10^9)     #    TypeError: unsupported operand ...
print(30.11 * 10 ** 9)      #      30110000000.0
```

Explanation:
You could replace the E by * 10 **

Question: 32

The ABC organics company needs a simple program
that their call center will use to enter survey data for a new coffee variety.
The program must accept input
and return the average rating based on a five-star scale.
The output must be rounded to two decimal places.
You need to complete the code to meet the requirements.

```
sum = count = done =  0
average = 0.0
while done !=  -1:
rating = XXX
if rating ==  -1:
break
sum += rating
count +=  1
average = float(sum / count)
YYY + ZZZ
```

What should you insert instead of XXX, YYY and ZZZ?

A.

```
XXX -> float(input('Enter next rating (1-5), -1 for done'))
YYY -> print('The average star rating for the new coffee is: ' ZZZ -> format(average,
'.2d'))
```

B.

```
XXX -> float(input('Enter next rating (1-5), -1 for done'))
YYY -> printline('The average star rating for the new coffee is:
ZZZ -> format(average, '.2f'))
```

C.

```
XXX -> print(input('Enter next rating (1-5), -1 for done'))
YYY -> print('The average star rating for the new coffee is: ZZZ -> format(average,
'.2f'))
```

D.

```
XXX -> float(input('Enter next rating (1-5), -1 for done'))
YYY -> output('The average star rating for the new coffee is: ZZZ -> format(average,
'.2d'))
```

E.

```
XXX -> float(input('Enter next rating (1-5), -1 for done'))
YYY -> print('The average star rating for the new coffee is: ZZZ -> format(average,
'.2f'))
```

F.

```
XXX -> input('Enter next rating (1-5), -1 for done')
YY -> print('The average star rating for the new coffee is: ZZ -> format(average,
'.2d'))
```

Answer: E

Explanation:

Topics: while not equal to operator if add and assign operator float() division operator format() plus operator string concatenation Try it yourself: `sum = count = done = 0`

```
average = 0.0
while done != -1:
rating = float(input('Enter next rating (1-5), -1 for done'))
if rating == -1:
break sum += rating count += 1
average = float(sum / count) print('The average star rating for the new coffee is: ' +
format(average, '.2f')) # format(average, '.2d')      -> ValueError: ...
```

Explanation:

The input() function always returns a string

You need to cast that string to a float with the float() function. The function to print something to the monitor is called print() And if you want to round a float to two decimal places, you need the format string '.2f'

Question: 33

Consider the following code snippet:

```
w = bool(23)
x  = bool('')
y = bool(' ')
z  = bool([False])
```

Which of the variables will contain False?

A. z
B. x
C. y
D. w

Answer: B

Explanation:

Topic: type casting with bool()

Try it yourself:

```
print(bool(23))       #  True
print(bool(''))       #  False
print(bool(' '))      #  True
print(bool([False])) #   True
```

Explanation:

The list with the value False is not empty and therefore it becomes True

The string with the space also contain one character

and therefore it also becomes True

The values that become False in Python are the following:

```
print(bool(''))          #False
print(bool(0))           #False
print(bool(0.0))         #False
print(bool(0j))      #   False
print(bool(None))    #   False
print(bool([]))          # False
print(bool(()))          # False
print(bool({}))      #   False
print(bool(set()))   # False
print(bool(range(0))) # False
```

Question: 34

What is the expected output of the following code?

```
def func(num):
res =  '*'
for _ in range(num): res += res return res
for x in func(2): print(x, end='')
```

A. **
B. The code is erroneous.
C. *
D. ****

Answer: D

Explanation:

Topics: def return for

Try it yourself:

```
def func(num): res =       '*'
for _ in range(num): res += res
return res
for x in func(2):
print(x, end='')     # ****
# print(x, end='-')  # *-*-*-*
print()
print(func(2)) #   ****
```

Explanation:

The for loop inside of the function will iterate twice.

Before the loop res has one star.

In the first iteration a second star is added.

res then has two stars.

In the second iteration two more stars are added to those two star and res will end up with four stars.

The for loop outside of the function will just iterate through the string and print every single star.

You could get that easier by just printing the whole return value.

Question: 35

What is the expected output of the following code?

```
num = 1
def func(): num = num +    3
print(num) func() print(num)
```

A. 4 1

B. 4 4

C. The code is erroneous.

D. 1 4

E. 1 1

Answer: C

Explanation:

Topics: def shadowing

Try it yourself:

```
num =  1
def func():
# num = num +  3  # UnboundLocalError: ...
print(num)
func()
print(num)
print('  --------  ')
num2 =  1
def func2():
x = num2 +  3
print(x) # 4
func2()
print('  ------  ')
num3 =  1
def func3():
num3 =  3     # Shadows num3 from outer scope
```

```
print(num3) #   3
func3()
```

Explanation:
A variable name shadows into a function.
You can use it in an expression like in func2() or you can assign a new value to it like in func3() BUT you can not do both at the same time like in func() There is going to be the new variable num and you can not use it in an expression before its first assignment.

Question: 36

The result of the following addition: `123      +      0.0`

A. cannot be evaluated
B. is equal to 123.0
C. is equal to 123

Answer: B

Explanation:
Topics: addition operator integer float
Try it yourself:

```
print(123 +  0.0) #  123.0
```

Explanation:
If you have an arithmetic operation with a float, the result will also be a float.

Question: 37

What is the expected output of the following code? `print(list('hello'))`

A. None of the above.
B. hello
C. [h, e, l, l, o]
D. ['h', 'e', 'l','l', 'o']
E. ['h' 'e' 'l' 'l' 'o']

Answer: D

Explanation:

Topic: list()

Try it yourself:

```
print(list('hello')) #   ['h', 'e', 'l', 'l', 'o']
```

Explanation:

A string is a sequence of characters and works very fine with the list() function.

The result is a list of strings, in which every character is a string of its own.

Question: 38

What is the default return value for a function that does not explicitly return any value?

A. int
B. void
C. None
D. Null
E. public

Answer: C

Explanation:

Topic: return

Try it yourself:

```
def func1():
pass
print(func1()) # None def func2(): return
print(func2()) # None
```

Explanation:

If a function does not have the keyword return the function will return the value None

The same happens if there is no value after the keyword return

Question: 39

Which of the following lines correctly invoke the function defined below:

```
def fun(a, b, c=0): # Body of the function.
```

(Select two answers)

A. fun(0, 1, 2) fun(b=0, a=0) fun(b=1) fun()
B.
C. **Answer: A,B**
D. **Explanation:**

Topics: functions positional parameters keyword parameters Try it yourself:

```
def fun(a, b, c=0): # Body of the function. pass fun(b=0, a=0) fun(0, 1,      2)
```

```
#   fun()   #   TypeError: fun() missing 2 required
# positional arguments: 'a' and 'b'
#   fun(b=1)   #          TypeError:      fun() missing 1
    required
# positional argument: 'a'
```

Explanation:
Only the parameter c has a default value.
Therefore you need at least two arguments.

Question: 40

What is the expected output of the following code?

```
x =  '\''
print(len(x))
```

A. 1
B. 2
C. The code is erroneous.
D. 0

Answer: A

Explanation:
Topics: len() escaping
Try it yourself:

```
print(len('\''))  # 1
```

Explanation:
The backslash is the character to escape another character.
Here the backslash escapes the following single quote character.
Together they are one character.

Question: 41

Which of the following statements are true?

A. (Select two answers)
B. The ** operator uses right-sided binding.
C. The result of the / operator is always an integer value.
D. The right argument of the % operator cannot be zero.
E. Addition precedes multiplication.

Answer: B,D

Explanation:
Topics: exponentiation/power operator right-sided binding
Try it yourself:

```
print(4 **  3 ** 2)    # 262144
print(4 **  (3 ** 2)) #  262144
print(4 **  9)         # 262144
print(262144)          # 262144
# print(7 %  0)  # ZeroDivisionError
```

Explanation:
If you have more than one power operators

next to each other, the right one will be executed first.
https://docs.python.org/3/reference/expressions.html#the-power-operator
To check the rest of a modulo operation,
Python needs to divide the first operand by the second operand.
And like in a normal division, the second operand cannot be zero.

Question: 42

What do you call a tool that lets you lanch your code step-by-step and inspect it at each moment of execution?

A. A debugger
B. An editor
C. A console

Answer: A

Explanation:
Topic: debugger
Explanation:
https://en.wikipedia.org/wiki/Debugger

Question: 43

What is the expected output of the following code?

```
list1 = [1, 3]
list2 = list1 list1[0] = 4
print(list2)
```

A. [1, 4]
B. [4, 3]
C. [1, 3, 4]
D. [1, 3]

Answer: B

Explanation:
Topics: list reference of a mutable data type
Try it yourself:

```
list1 = [1, 3]
list2 = list1 list1[0] = 4
print(list2) # [4, 3]
print(id(list1)) # e.g. 140539383947452
print(id(list2)) # e.g. 140539383947452 (the same number)
```

Explanation:
A list is mutable.
When you assign it to a different variable, you create a reference of the same object.
If afterwards you change one of them, the other one is changed too.

Question: 44

How many stars will the following code print to the monitor?

```
i = 0
```

```
while i <=  3:
i +=  2
print('*')
```

A. one
B. zero
C. two
D. three

Answer: C

Explanation:

Topic: while

Try it yourself:

```
i =  0
while i <=  3:    # i=0,i=2
i +=  2
print('*')
"""

*
*
```

Explanation:
In the first iteration of the while loop i is 0
i becomes 2 and the first star is printed.
In the second iteration of the while loop i is 2
i becomes 4 and the second star is printed.
i is 4 and therefore 4 <= 3 is False what ends the while loop.

Question: 45

What is the expected output of the following code if the user enters 2 and 4?

```
x = input()  y = input()
print(x + y)
```

A. 4
B. 6
C. 24
D. 2

Answer: C

Explanation:

Topics: input() plus operator string concatenation Try it yourself:

```
E.   x = input()
F.   y = input()
x, y =  '2','4'  # Just for convenience
print(x + y)      #  24
```

Explanation:
As always the input() function return a string.
Therefore string concatenation takes place and the result is the string 24

Question: 46

What will be the output of the following code snippet?

```
x = 2
y = 1
x *= y + 1
print(x)
```

A. 3
B. 4
C. None
D. 2
E. 1

Answer: B

Explanation:

Topics: addition operator multiply and assign operator operator precedence

Try it yourself:

```
x = 2
y = 1
F. x *= y + 1
x = x * (y + 1)
print(x) #  4
```

Explanation:

The operator precedence of the addition operator is higher than the operator precedence of the multiply and assign operator That means the addition takes place before the multiplication.

Question: 47

What is the output of the following snippet?

```
dictionary = {'one': 'two', 'three': 'one', 'two': 'three'}
v = dictionary['one']
for k in range(len(dictionary)):
v = dictionary[v]
print(v)
```

A. three
B. ('one', 'two', 'three')
C. one
D. two

Answer: D

Explanation:

Topics: dictionary for range()

Try it yourself:

```
dictionary = {'one': 'two', 'three': 'one', 'two': 'three'}
v = dictionary['one']
# for k in range(len(dictionary)):
for k in range(3) :
print(v)
```

```
G.   two
H.   three
I.   one
v = dictionary[v]
print(v) # two
```

Explanation:
Before the for loop the value of index 'one' is assigned to v
And that value is 'two'
In the first iteration of the for loop the value of index 'two' is assigned to v
And that value is 'three'
In the second iteration of the for loop the value of index 'three' is assigned to v And that value is 'one'
In the third iteration of the for loop the value of index 'one' is assigned to v
And that value is again 'two'
After the for loop that value 'two' is printed.

Question: 48

What is the expected output of the following code?
```
x =  1      /
     2 + 3 // 3 +  4 **2
print(x)
```

A. 8.5
B. 8
C. 17.5
D. 17

Answer: C

Explanation:
Topics: arithmetic operators operator precedence
Try it yourself:

```
x = 1 / 2 + 3 / 3 +  4 ** 2
print(x)         /                  # 17.5
print(1 / 2 + 3 / 3  + 4 ** 2)      # 17.5
print(1 / 2 + 3 / 3  + (4 ** 2))    # 17.5
print(1 / 2 + 3 / 3  + 16)          # 17.5
print((1   2)  + 3 // 3 + 16)       # 17.5
print(0.5 + 3 / 3 +  16)            # 17.5
print(0.5 +    //   3) + 16)       # 17.5
print(0.5 + 1 + 16)                 # 17.5
print((0.5 + 1)  +                  # 17.5
print(1.5 + 16)    16               # 17.5
print(17.5)       )                 # 17.5
```

Explanation:
The operators here come from three different groups: "Exponent" has the highest precedence.
Followed by "Multiplication, Division, Floor division, Modulus".
"Addition, Subtraction" has the lowest precedence.
Therefore the order of operations here is: ** -> / -> // -> + -> +

Question: 49

If a list passed into a function as an argument, deleting any of its elements inside the function using the del instruction:

A. will cause a runtime error
B. will not affect the argument
C. will affect the argument

Answer: C

Explanation:

Topics: pass by reference

Try it yourself:

```
my_list =  [1, 2, 3]
def delete_first(x):
del x[0]
delete_first(my_list)
print(my_list) #   [2, 3]
```

Explanation:
A list is a mutable data type and it is pass by reference to a function.
Meaning that the list inside of the function
and the list outside of the function
will point to the same object in the memory.
If you change the list inside of the function
that will change the list outside of the function in the same way.

Question: 50

An operator able to check whether two values are not equal is coded as:

A. not ==
B. =/=
C. !=
D. <>

Answer: C

Explanation:

Topic: not equal to operator

Try it yourself:

```
print(1 !=  2)          # True
# print(1  <> 2)      #  SyntaxError:...
# print(1  =/=2)      #  SyntaxError:...
# print(1  not==  2)  #  SyntaxError:...
```

Explanation:
Other languages have <> or =/= as not equal to operators
In Python the not equal to operator is != not == does not work like this,
because you can not have two operators next to each other. It would work
like that: print(not(1 == 2)) # True

Question: 51

What is the expected output of the following code? `def fun(): return`

```
True x = fun(False) print(x)
```

A. False
B. 0
C. 1
D. The program will cause an error
E. True

Answer: D

Explanation:

Topics: functions parameter/argument return boolean

Try it yourself:

```
def fun(): return True x = fun(False)
# TypeError: fun() takes 0 positional arguments but 1 was given
print(x)
```

Explanation:
The fun() function is defined with any parameters and therefore it cannot be called with an argument.

Question: 52

What will be the output of the following code snippet?

```
a =  [1, 2, 3, 4, 5, 6, 7, 8, 9]
print(a[::2])
```

A. [1, 2]
B. [1, 3, 5, 7, 9]
C. [8, 9]
D. [1, 2, 3]

Answer: B

Explanation:
Topic: list slicing
Try it yourself:

```
a =  [1, 2, 3, 4, 5, 6, 7, 8, 9]
print(a[::2]) #  [1, 3, 5, 7, 9]
```

Explanation:
List slicing: [start(inclusive):end(exclusive):step]
The start and end values are missing here.
If you leave them out, you will slice from the beginning to the end.
If you leave the end out it will also be inclusive.
The third value (the step) is 2 therefore every second element gets taken.

Question: 53

What value will be assigned to the x variable?

```
z =  3
y =  7
x = y < z and z > y or y > z and z < y
```

A. 0
B. False
C. 1
D. True

Answer: D

Explanation:
Topic: greater than operator less than operator logical and operators logical or operator operator precedence
Try it yourself:

```
z =  3
y = 7
x = y < z and z > y or y > z and z < y
```

```
print(x)
print(y < z and z > y or y > z and z < y)
print(7 < 3 and 3 > 7 or 7 > 3 and 3 < 7)
print(False and False or True and True)
print((False and False) or (True and True))
print(False or True) print(True)            # True # True # True # True # True # True # True
```

Explanation:
The operators here are from three different groups.
"Comparisons, Identity, Membership operators", "Logical AND", "Logical OR".
The two comparison operators
the greater than operator and the less than operator
have the highest precedence.
Then the logical and operator has a higher precedence
than the logical or operator

Question: 54

What is the expected output of the following code?

```
x = 1
if x > 0 or x < 1:
print("1") if x >1:
print("2") elif x >= 1:
print("3") else:
print("4")
```

A.

```
1
3
```

B.

```
1
3
4
```

C.

```
1
```

D.

```
1 4
```

Answer: A

Explanation:

Topics: if elif else logical operators comparison operators

Try it yourself:

```
x = 1
if x > 0 or x < 1:
print("1") # 1 if x >  1:
print("2") elif x >= 1: print("3") # 3 else:
print("4")
```

Explanation:
1 is greater than 0 and therefore the condition
of the first if evaluates to True and 1 is printed.
1 is greater than or equal to 1 and therefore the condition of the elif also evaluates to True and 3 is also printed.

Question: 55

What is the expected output of the following code?

```
x  = 28
y  = 8
print(x / y)
print(x // y) print(x % y)
```

A.

```
3
3.5
4
```

B.

```
3.0
3
2
```

C.

```
3.5
3
4
```

D.

```
3.5
3.5
2
```

Answer: C

Explanation:
Topic: arithmetic operators
Try it yourself:

```
a  = 28
b  = 8
print(a  / b)   #3.5
print(a  // b)  #3
print(a  % b)   #4
```

Explanation:
Everything normal here.
The division operator the floor division operator and the modulus operator all do their normal job.

Question: 56

What is the output of the following code snippet?

```
def test(x=1, y=2):
x = x + y
y +=  1
print(x, y)
test(2, 1)
```

A. 2 3
B. 3 2
C. 1 3
D. 3 3
E. The code is erroneous.

Answer: B

Explanation:
Topics: def parameter
Try it yourself:

```
def test(x=1, y=2):
x  = x + y   #   2 + 1 -> 3
y  += 1      #   1 + 1 -> 2
print(x, y) #   3 2
test(2, 1)   #  3 2
```

Explanation:
Okay, both parameters get the default value of the other one, but for the rest it's business as usual.

Question: 57

Which of the following variable names is illegal?

A. in
B. In
C. IN
D. in

Answer: A

Explanation:
Topic: naming variables
Try it yourself:

```
# in = 'Hello' # SyntaxError: invalid syntax
#  Does not work because "in" is a python keyword
#  for the membership operator:
print(7 in [1,  4, 7, 11])  # True
```

```
#  Those work because python is case sensitiv
In = 'Hello'
IN = 'Hello'
#  This one works because the underscore
#  is a valid character for naming variables:
in = 'Hello'
```

Explanation:
You can not name a variable like a Python keyword.
Here is a list of all the Python keywords:

```
import keyword
print(keyword.kwlist)
"""
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await',
'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except',
'finally', 'for', 'from', 'global', 'if', 'import', 'in',   'is',
'lambda', 'nonlocal', 'not', 'or',
'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

Here are the other rules for naming variables in Python:
A variable name must start with a letter or the underscore character.
A variable name can not start with a number.
A variable name can only contain alpha-numeric characters
and underscores (a-z, 0-9, and _)
Variable names are case-sensitive
(age, Age and AGE are three different variables)

Question: 58

What is the expected output of the following code?

```
x =  1 // 5 + 1  / 5
print(x)
```

A. 0.2
B. 0
C. 0.4
D. 0.0

Answer: A

Explanation:
Topics: arithmetic operators operator precedence
Try it yourself:

```
x =  1 // 5 + 1  /  5
x =  (1 // 5) +  (1 / 5)
print(1 //  5)   # 0
x =  0 +  (1/ 5)
x =  0 +  0.2
print(x) #   0.2
```

Explanation:
The operators here come from two different groups:
The group "Multiplication, Division, Floor division, Modulus" has a higher precedence than the group "Addition, Subtraction".
Therefore the order of operations here is: // -> / -> +

Question: 59

isalnum() checks if a string contains only letters and digits, and this is:

A. Amethod
B. Amodule
C. Afunction

Answer: A

Explanation:
Topic: isalnum()
Try it yourself:

```
print('James007'.isalnum())     # True
print('Hello world'.isalnum())  # False
```

Explanation:
isalnum() is a String-Method.
https://www.w3schools.com/python/ref_string_isalnum.asp
'Hello world' is not alphanumeric, because of the space character.

Question: 60

The function body is missing.
What snippet would you insert in the line indicated below:

```
def func(number):
# insert your code here print(func(7))
```

A. print(number) B. return number
D. return 'number'
E. print('number') **Answer: B** **Explanation:** Topics: def return Try it yourself:

```
def func(number):
return number     #  worksand only 7 would be printed
#  print(number)   #  works,but 7 & None would be printed
#  print('number') #  works,but number & None would be printed
#  return 'number' #  works,but number would be printed
print(func(7))
```

Explanation:
The parameter name is number therefore it can not be in quotes.
If you only print something in the function,
the function will return None and that is not wanted here, because the return values gets already printed outside of the function.

Question: 61

Which of the approachable except: branches
is taken into consideration when an exception occurs?

A. The first matching branch.
B. The last matching branch.

C. Any of the matching branches.

Answer: A

Explanation:

Topic: except

Try it yourself:

```
try: zahl = 100      /      0
except ArithmeticError: print('ArithmeticError') # ArithmeticError
except ZeroDivisionError:
print('ZeroDivisionError') try: zahl =  100    /      0
except ZeroDivisionError: print('ZeroDivisionError') #
ZeroDivisionError except ArithmeticError: print('ArithmeticError')
print(issubclass(ZeroDivisionError, ArithmeticError)) # True
```

Explanation:

The ZeroDivisionError is a subclass of the ArithmeticError Therefore here they are both approachable except: branches. And the first match is taken into consideration.

Question: 62

What is the expected output of the following code?

```
nums =  [3, 4, 5, 20, 5, 25, 1, 3]
nums.pop(1)
print(nums)
```

A. [3, 1, 25, 5, 20, 5, 4]
B. [1, 3, 3, 4, 5, 5, 20, 25]
C. [3, 4, 5, 20, 5, 25, 1, 3]
D. [1, 3, 4, 5, 20, 5, 25]
E. [3, 5, 20, 5, 25, 1, 3]

Answer: E

Explanation:

Topics: pop() list

Try it yourself:

```
nums =  [3,  4, 5, 20,  5,  25, 1, 3]
nums.pop(1)
print(nums) #   [3, 5, 20, 5, 25, 1, 3]
```

Explanation:

list.pop([i])

The index is optional.

If the index is given, pop() removes and returns the element at the given index.

The default index is -1

Meaning that the last index is removed and returned.

Here the index 1 gets removed: the number 4

Question: 63

The 0o prefix means that the number after it is denoted as:

A. decimal
B. hexadecimal
C. binary
D. octal

Answer: D

Explanation:

Topic: octal notation

Try it yourself:

```
print(0o10)   # 8
print(0o77) # 63
```

Explanation:

The octal numeral system, or oct for short, is the base-8 number system, and uses the digits 0 to 7

Question: 64

Insert the correct snippet so that the program produces the expected output.

Expected output:

```
True
```

Code:

```
list = [False, True, "2", 3, 4, 5]
E.  insert code here
print(b)
```

A. b =False
B. b =0 in list
C. b =0 notin list
D. b =list[0]

Answer: B

Explanation:

Topics: list in operator implicit type casting

Try it yourself:

```
list = [False, True, "2", 3, 4, 5]
b =  0 in list
print(b) # True
# The same without the in operator: list2 = [False, True, "2",     3,
                            4, 5]
res = False
for i in list2:
if i ==  0:
res = True
print(res) # True
print(0 == False) # True
```

Explanation:

The in operator uses the equal to operator on every element of the list.
The value 0 is not in the list as a literal,
BUT because 0 == False evaluates to True
the False element makes it look like 0 would be in the list.

Question: 65

What is the expected output of the following code?

```
x = 1
y = 2
x, y,   z =x, x,  y
z, y,   z=x, y,  z
print(x, y, z)
```

A. 2 12
B. 1 12
C. 1 21
D. 1 22

Answer: B

Explanation:

Topic: multiple assignments

Try it yourself:

```
x =  1
y =  2
#  x, y, z   =   x, x, y
x, y, z =   1,  1,  2   #  ->   x=1;y=1;   z=2
#  z, y, z   =   x, y, z
z, y, z =   1,  1,  2   #  ->   y=1;z=2
#  z will be 1 first and then become 2
#  x is still 1
print(x, y, z)      #  1 1 2
```

Explanation:

We have multiple assignment in Python.

You can assign multiple values to multiple variables.

```
a, b =  3, 7
```

It is just shorter than

```
a =  3
b =  7
```

It does not have practical use, but what also works is `c, c =  23,  42`

First c becomes 23 and then c becomes 42

You better write `c =  42`

The same happens in this question in `z, y, z =  1,  1,  2`

First z becomes 1 and then z becomes 2

You can also assign the same value to multiple variables: `a = b = c = d =`

```
                1
print(a, b, c, d) #  1 1 1 1
```

Question: 66

Which of the following statements is false?

A. The Nonevalue may not be used outside functions.

B. The Nonevalue can be assigned to variables.
C. The Nonevalue can not be used as an argument of arithmetic operators.
D. The Nonevalue can be compared with variables.

Answer: A

Explanation:

Topic: None

Try it yourself:

```
#  The None value can be assigned to variables.
x = None
print(x) # None
#  The None value can be compared with variables.
y =  3
print(y is None)    # False
#  The None value cannot be used
#  as an argument of arithmetic operators:
#  print(None + 7)  # TypeError: unsupported operand ...
```

Explanation:

It is true that the None value can not be used as an argument of arithmetic operators.

But the None value can be assigned and compared to variables. And that can absolutely happen outside of a function.

Question: 67

The ** operator:

A. performs duplicated multiplication
B. performs exponentiation
C. does not exist
D. performs floating-point multiplication

Answer: B

Explanation:

Topic: exponentiation/power operator

Try it yourself:

```
print(2 **  8) #  256
print(256)     #  256
```

Explanation:

https://docs.python.org/3/reference/expressions.html#the-power-operator

Question: 68

What is the expected output of the following code?

```
data =  {}
def func(d, key, value): d[key] = value
print(func(data, '1', 'Peter'))
```

A. None
B. 1
C. Peter

D. value
E. The code is erroneous.

Answer: A

Explanation:

Topics: def return

Try it yourself:

```
data = {}
def func(d, key, value):
d[key] = value
print(func(data, '1', 'Peter')) # None
```

Explanation:

func() has no return statement.

Therefore None gets returned.

Question: 69

What value will be assigned to the x variable?

```
z = 3
y = 7
x = y == z and y > z or z > y and z != y
```

A. 1
B. False
C. True
D. 0

Answer: B

Explanation:

Topics: logical operators relational operators operator precedence

Try it yourself:

```
z = 3
y = 7
x = y == z and y > z or z > y and z != y      # False
print(x)                                      # False
print(7 == 3 and 7 > 3 or 3 > 7 and 3 != 7)   # False
print(False and True or False and True)       # False
print((False and True) or (False and True)) # False print(False or
False)                                        # False
print(False)                                  # False
```

Explanation:

The operators here are from three different groups.

"Comparisons, Identity, Membership operators", "Logical AND", "Logical OR".

The three different comparison operators greater than operator equal to operator not equal to operator have the highest precedence. Then the logical and operator has a higher precedence than the logical or operator

Question: 70

What is the expected output of the following code? `print(1 / 1)`

A. 1.0

B. This can not be predicted.
C. 1
D. This can not be evaluated.

Answer: A

Explanation:

Topic: division operator

Try it yourself: `print(1 / 1) # 1.0`

Explanation:

All you have to know here is, that the division operator always returns a float.
Even if it is operating with two integers.

Question: 71

UTF-8 is ...

A. a synonym for "byte".
B. the 9th version of the UTF Standard.
C. a Python version name.
D. an encoding form of the Unicode Standard.

Answer: D

Explanation:

Topic: UTF-8

Explanation:

UTF-8 is one of three encoding forms of the Unicode Standard.
The others are UTF-16 and UTF-32.
UTF-8 is the most popular one, because it is most flexibel.
UTF-8 requires 8, 16, 24 or 32 bits (one to four bytes) to encode a Unicode character,
UTF-16 requires either 16 or 32 bits to encode a character, and UTF-32 always requires 32 bits to encode a character.

Question: 72

What is the expected output of the following code?

```
data = {'1': '0', '0': '1'}
for d in data.vals(): print(d, end=' ')
```

A. The code is erroneous.
B. 0 1
C. 0 0
D. 1 0

Answer: A

Explanation:

Topics: dictionary values() for

Try it yourself:

```
data = {'1': '0', '0': '1'}
E. for d in data.values():
for d in data.vals():  # AttributeError: ...
print(d, end=' ')    #  0 1
```

Explanation:
This question is a little tricky.
Everything seems to be fine,

BUT the right name of the dictionary method is values()

Question: 73

What is the expected output of the following code?

```
print(float('1.3'))
```

A. The code is erroneous.
B. 13
C. 1,3
D. 1.3

Answer: D

Explanation:
Topic: float()
Try it yourself:

```
print(float('1.3')) #   1.3
```

Explanation:
The function float() does its normal job here and converts the string to a float.

Question: 74

How many stars will the following code print to the monitor?

```
x = 1
while x < 10:
print('*')
x = x << 1
```

A. four
B. eight
C. two
D. one

Answer: A

Explanation:
Topics: while bit operator
Try it yourself:

```
x = 1
while x < 10:
print('*')
x = x << 1
print('x:', x)              #   2 -> 4 -> 8 -> 16
print('bin(x):', bin(x)) # 0b10->0b100->0b1000->0b10000 """ 1      ->
00001  <<   1->00010 (2)
2  ->  00010<<1 -> 00100  (4)
4  ->  00100<<1 -> 01000  (8)
8  ->  01000<<1 -> 10000  (16)
```

Explanation:
Left shift by one, a classic way to double values.
Every value goes to the bit on its left and thereby doubles in value.
And 1, 2, 4, 8 are all smaller than 10 but not the 16 and therefore four stars will be printed.

Question: 75

How many stars will the following snippet print to the monitor?

```
i = 0
while i <= 5:
i += 1
if i % 2 == 0:
break
print('*')
```

A. zero
B. three
C. one
D. two

Answer: C

Explanation:

Topics: while break

Try it yourself:

```
i = 0
while i <= 5:
print('1. i: ', i) # 0 -> 1
i += 1
print('2. i: ', i) # 1 -> 2
if i % 2 == 0: # False, True
break print('*') # *
```

Explanation:
i % 2 == 0 is known as test for even numbers.
In the first iteration i is an odd 1
and therefore the break does not get triggered
and a star gets printed.
In the second iteration i is an even 2
the break gets triggered and there is no more stars.

Question: 76

Which of the following variable names is illegal?

A. True
B. tRUE
C. TRUE
D. true

Answer: A

Explanation:

Topic: naming variables

Try it yourself:

```
# True =3 # SyntaxError: cannot assign to True
# Those three work, because Python is case sensitive: true = 7
tRUE = 23
TRUE = 42
print(true, tRUE, TRUE) # 7 23 42
```

Explanation:

You can not name a variable like a Python keyword.

Question: 77

What is the expected output of the following code?

```
x = '\\'
print(len(x))
```

A. 1
B. 4
C. 2
D. The code is erroneous.

Answer: C

Explanation:
Topics: escaping len()
Try it yourself:

```
# print(len('\'))    # SyntaxError: ...
print(len('\'))      # 1
# print(len('\\')) # SyntaxError: ...
print(len('\\'))     # 2
```

Explanation:
The backslash is the character to escape another character.
If you write '\' the backslash escapes
the ending single quote to a normal character.
It takes its syntactical meaning
and the single quote becomes a normal character and it looses its ability to end the string.
And therefore we get the syntax error.
If you write '\' the one backslash escapes the other and you end up with a string with one normal backslash.
If you write '\\' it is kind of the same and you end up with a string with two normal backslashes.

Question: 78

What is the expected output of the following code?

```
data = 'abcdefg' def func(text): del text[2] return text
print(func(data))
```

A. abcef
B. The code is erroneous.
C. acdef
D. abdef

Answer: B

Explanation:
Topics: def del string (immutable) string indexing
Try it yourself:

```
data = 'abcdefg' def func(text):
#  del text[2] # TypeError: ...
```

```
return text
print(func(data))
#  Strings are immutable
#  The indexes are readable but not writeable: s = 'Hello'
print(s[0])   # H
#  s[0] = 'h' # TypeError: ...
```

Explanation:
A string is immutable. You can not change it.
You can read something by indexing,
BUT you can not write something by indexing.

Question: 79

What is the expected output of the following code?

```
data = [[0,   1,     2,
        3] for i in range(2)]
print(data[2][0])
```

A. The code is erroneous.
B. 1
C. 2
D. 0

Answer: A

Explanation:
Topics: list comprehension list indexing IndexError
Try it yourself:

```
data =  [[0,  1,   2,      3] for i in range(2)]
print(data[2][0]) # IndexError: list index out of range
```

```
print(data)      # [[0,   1, 2,    3],    [0,    1,     2,     3]]
print(data[0])   #   [0,    1,     2,     3]
print(data[1])   #   [0,    1,     2,     3]
```

Explanation:
range(2) has two elements: 0 and 1
Therefore the outer list will have two elements.
And data[2] does not exist.

Question: 80

How many stars will the following code send to the monitor?

```
x =  0
while x <  6:
x +=  1
if x %  2 == 0:
continue print('*')
```

A. two
B. one
C. three
D. zero

Answer: C

Explanation:
Topics: while if continue
Try it yourself:

```
x =  0
while x <  6:
print('1.  x:', x)          # 0 -> 1 -> 2 -> 3 -> 4 -> 5
x +=  1
print('2.  x:', x)          # 1 -> 2 -> 3 -> 4 -> 5 -> 6
if x %  2 == 0:
print('x in if:', x) #   2 -> 4 -> 6
continue
print('x behind if:', x) #   1 -> 3 -> 5
print('*') """
*
*
*
```

Explanation:
When x is 2, 4 and 6 the if condition is True
and the continue gets triggered.
The iteration ends directly and the print() function doesn't get executed.
When x is 1, 3 and 5 the if condition is False
the continue does not get triggered
and those three times the print() function gets executed.
Therefore there will be three stars printed.

Question: 81

Which of the following function calls can be used to invoke the below function definition?

```
def test(a, b, c, d):
```

Choose three.

A. test(a=1, b=2, c=3,d=4)
B. test(1, 2, 3, d=4)
C. test(a=1, 2, c=3, 4)
D. test(a=1, b=2, c=3, 4)
E. test(1, 2, 3, 4)
F. test(a=1, 2, 3, 4)

Answer: A,B,E

Explanation:

Topics: def positional/keyword arguments

Try it yourself:

```
def test(a, b, c, d):
print(a, b, c, d)
test(1, 2, 3, 4)          # 1 2 3 4
test(a=1, b=2, c=3,d=4)   # 1 2 3 4
test(1, 2, 3, d=4)        # 1 2 3 4
# test(a=1, 2, 3, 4).     # SyntaxError: ...
# test(a=1, b=2, c=3,     # SyntaxError: ...
  4)                      # SyntaxError: ...
# test(a=1, 2, c=3, 4)    # SyntaxError: ...
# print(end='', 'Hello')
```

Explanation:

The keyword arguments always have to be at the end.

Question: 82

What is the decimal value of the following binary number? `1010`

A. 8
B. 10
C. 12
D. 4

Answer: B

Explanation:

Topics: binary numbers

Try it yourself:

```
print(0b1010)   #  10
print(bin(10))  #  0b1010
print(0b0010)   #  2
print(0b1000)   #  8
print(0b1000 +  0b0010) #  10
```

Explanation:

The value of the second bit is 2 and the value of the forth bit is 8 That is 10 in total.

Question: 83

How did Python, the programming language, get its name?

A. Guido van Roddum named it to honor Python of Catana, a dramatic poet of the time of Alexander the Grerat
B. Guido van Roddum named it after the Pythonidae - a family of large, nonvenomous snakes
C. Guido van Roddum named it to honor Monty Python's Flying Circus, a BBC comedy series popular in the 1970s

Answer: C

Explanation:
Topic: Python's name
Explanation:
While you may know the python as a large snake,
the name of the Python programming language comes from
an old BBC television comedy sketch series
called Monty Python's Flying Circus.
At the height of its success,
the Monty Python team were performing their sketches to live audiences across the world, including at the Hollywood Bowl.
Since Monty Python is considered one of the two
fundamental nutrients to a programmer (the other being pizza), Python's creator named the language in honor of the TV show.
https://en.wikipedia.org/wiki/Python_(programming_language)#Design_philosophy_and_features

Question: 84

Which of the following sentences correctly describes the output of the below Python code?

```
data =  [4, 2, 3, 2, 1]
res = data[0]
for d in data:
if d < res: res = d
print(res)
```

A. res is the largest number in the list.
B. None of the above.
C. res is the average of all the number in the list.
D. res is the smallest number in the list.
E. res is the sum of all the number in the list.

Answer: D

Explanation:
Topics: for if list
Try it yourself:

```
data =  [4, 2, 3, 2, 1]
res = data[0] for d in data: if d < res:
print('d in if:', d) # 2   -> 1
res = d
print(res) #   1
```

Explanation:
Classic way the find the smallest number.
Take the first element as possible result.
Compare the next number with the possible result. If the next number is smaller, it becomes the new possible result and so forth. In the end the result is the smallest number.

Question: 85

What is the expected output of the following code if the user enters 11 and 4

```
x = int(input()) y = int(input()) x = x % y
x = x % y y = y % x print(y)
```

A. 1
B. 2
C. 3
D. 4

Answer: A

Explanation:
Topics: input() modulus operator
Try it yourself:

```
#  x = int(input())  # Input: 11
#  y = int(input())  # Input: 4
x, y =  11, 4    # Just for convenience
x = x %  y
print(11   %  4) #3
x = x %  y
print(3  %4)    #3
y = y % x
print(4  %3)    #1
print(y)        #1
```

Explanation:
input() returns strings, but the int() function casts them to integers.
Then you have to concentrate with all the modulus operators.

Question: 86

What is the expected output of the following code?

```
x =  True
y = False x   =      x      or     y
y =  x andy
x =  x ory
print(x, y)
```

A. True False
B. False False
C. False True
D. True True

Answer: A

Explanation:
Topics: logical operators boolean
Try it yourself:

```
x = True
y = False
x =  x or y  # True  or False -> True
y =  x and y # True  and False ->False
x =  x or y  # True  or False -> True
print(x, y) # True False
```

Explanation:
Nothing changes here.
The variables always get assigned the value they had before.

Question: 87

The += operator, when applied to strings, performs:

A. Concatenation
B. Multiplication
C. Subtraction

Answer: A

Explanation:
Topic: add and assign operator
Try it yourself:

```
text = 'Hello'
text +=  ' '
text += 'World'
print(text)                    # Hello World
print('Hello' +  '  '      + 'World') # Hello World
```

Explanation:
The add and assign operator when applied to strings, performs a string concatenation.
Just like the addition operator

Question: 88

How many stars will the following code print to the monitor?

```
i =  0
while i < i +  2:
i +=  1
print('*') else:
print('*')
```

A. zero
B. two
C. one
D. The snippet will enter an infinite loop.

Answer: D

Explanation:
Topics: while else (nobreak)
Try it yourself:

```
i =  0
while i < i +  2:
i +=  1
print('*')
if i ==  1000: break # Safeguard
```

```
else: print('*')
```

Explanation:
i gets incremented inside the while loop, BUT i will always be smaller than i + 2 Therefore the while condition will always be True and we build ourselves a nice infinite loop.

Question: 89

Take a look at the snippet
and choose one of the following statements which is true:

```
nums =  []
vals = nums[:] vals.append(1)
```

A. vals is longer than nums
B. nums and vals are of the same length
C. nums is longer than vals

Answer: A

Explanation:
Topics: list slicing list copying list.append()
Try it yourself:

```
nums =  []
vals = nums[:] vals.append(1) print(nums)      # []
print(vals)  # [1]
```

Explanation:
nums[:] slices the list from beginning to end and therefore makes a copy.
If you change vals afterwards, nums will not also change.

Question: 90

What do you call a file containing a program written in a high-level programming language?

A. A target file
B. A code file
C. A source file
D. A machine file

Answer: C

Explanation:
Topics: high-level programming language source file
Explanation:
https://en.wikipedia.org/wiki/Source_code
https://en.wikipedia.org/wiki/Python_(programming_language)

Question: 91

What is the expected output of the following code?

```
data =  set([1, 2, 2, 3, 3, 3, 4, 4, 4, 4])
print(len(data))
```

A. 2
B. 0
C. 10
D. 1
E. 3
F. 4

Answer: F

Explanation:

Topics: list set() len()

Try it yourself:

```
data =  set([1, 2, 2, 3, 3, 3, 4, 4, 4, 4])
print(len(data)) #   4
print(type(data)) # <class 'set'>
print(data) # {1, 2, 3, 4}
# Also works directly:
print(len({1, 2,  2, 3, 3, 3, 4, 4, 4, 4})) #  4
```

Explanation:

A set can not have duplicates.

Question: 92

The following snippet:

```
def function_1(a):
return None
def function_2(a):
return function_1(a) * function_1(a)
print(function_2(2))
```

A. will output 4
B. will output 2
C. will cause aruntime error
D. will output 16

Answer: C

Explanation:

Topics: functions return None multiplication operator TypeError

Try it yourself:

```
def function_1(a): return None
def function_2(a):
return function_1(a) * function_1(a)
#  TypeError: unsupported operand type(s) for *:
#  NoneType' and 'NoneType'
print(function_2(2))
print(None * None)
#  TypeError: unsupported operand type(s) for *:
#  'NoneType' and 'NoneType'
print(function_2(2))
```

Explanation:

function_1() returns the None value

and any arithmetic operation with the None value will cause a TypeError.

Question: 93

What is the expected output of the following code? `data = 'abbabadaadbbaccabc'`

```
print(data.count('ab', 1))
```

A. 3
B. 5
C. 4
D. 2

Answer: D

Explanation:

Topics: string count()
Try it yourself:

```
data = 'abbabadaadbbaccabc' print(data.count('ab', 1))      # 2
print(data.count('ab', 0))  # 3
```

Explanation:
Remember to bring your best reading glasses to the exam.
The second parameter will determine where the count() method will start its counting.
The first parameter (in this case ab) is what count() will look for. count() will start at index 1
and therefore it will not find the ab right at the beginning of the string. That leaves two more ab to find.

Question: 94

What is the output of the following snippet?

```
def fun(in=2, out=3): return in * out
print(fun(3))
```

A. the snippet is erroneous (invalid syntax)
B. 9
C. 6

Answer: A

Explanation:

Topic: keyword parameters
Try it yourself:

```
#  def fun(in=2, out=3):
#    return in * out
def fun(inp=2, out=3): return inp * out
print(fun(3))
```

Explanation:
in is a keyword and therefore can not be used as a variable or a parameter.

Question: 95

A function definition:

A. may be placed anywhere inside the code after the first invocation
B. must be placed before the first invocation
C. cannot be placed among other code

Answer: B

Explanation:

Topic: functions NameError
Try it yourself:

```
#  my_first_function()
#  NameError: name 'my_first_function' is not defined
def my_first_function():
print('Hello')
```

Explanation:
The interpreter works line by line and therefore
a function definition must be placed before the first invocation.

Question: 96

What is the expected output of the following code?

```
x =  8
y =  10
result = x //  3 * 3 / 2 + y % 2 ** 2
print(result)
```

A. 7.0
B. 6.0
C. 5.0
D. 5

Answer: C

Explanation:
Topics: arithmetic operators operator precedence
Try it yourself:

```
x =  9
y =  12
result = x //  2 * 2 / 2 + y % 2 ** 3
print(result) # 8.0 print(x // 2 * 2 / 2 + y % 2 ** 3) # 8.0 print(9 // 2 * 2 / 2 + 12 % 2 **
3) # 8.0 print(9 // 2 * 2 / 2 + 12 % (2 ** 3)) # 8.0 print(9 // 2 * 2 / 2 + 12 % 8) # 8.0
print((9 // 2) * 2 / 2 + 12 % 8) # 8.0 print(4 * 2 / 2 + 12 % 8) # 8.0 print((4 * 2) / 2 + 12
% 8) # 8.0 print(8 / 2 + 12 % 8) # 8.0 print((8 / 2) + 12 % 8) # 8.0 print(4.0 + 12 % 8) # 8.0
print(4.0 + (12 % 8)) # 8.0 print(4.0 + 4) # 8.0 print(8.0) # 8.0
```

Explanation:
The operators here are from three different groups:
"Exponent" has the highest precedence.
Followed by "Multiplication, Division, Floor division, Modulus".
"Addition, Subtraction" has the lowest precedence.
Therefore the order of operations here is: ** -> // -> * -> / -> % -> +

Question: 97

Which of the following items are present in the function header?

A. function name
B. parameter list
C. function name and parameter list
D. return value

Answer: C

Explanation:
Topics: def parameter function header
Try it yourself:

```
def func(para1, para2):  # This line is the function header
para1 + para2
```

Explanation:
In the function header is the keyword def the function name and the parameter list.

Question: 98

Only one of the following statements is true - which one?

A. multiplication precedes addition
B. addition precedes multiplication
C. neither statement can be evaluated

Answer: A

Explanation:
Topics: operator precedence
Try it yourself:

```
print(2  + 3 * 4)    #  14
print(2  + (3 *4))   #  14
print(2  + 12)       #  14
print(14)            #  14
```

Explanation:

Priority	Operator	
1	+ , -	unary
2	**	
3		
4	+ , -	binary

Question: 99

What is the output of the following code?

```
a = 10
b = 20
c = a > b
print(not(c))
```

A. The program will cause an error
B. True
C. False
D. None

Answer: B

Explanation:

Topics: not operator boolean greater than operator Try it yourself:

```
a = 10
b = 20
c = a > b print(c)  # False
print(not c) # True
```

Explanation:
c becomes False because 10 is not greater than 20 And not False is True

Question: 100

What code would you insert instead of the comment to obtain the expected output? Expected output:

```
a b c
```

Code:

```
dictionary = {}
my_list = ['a', 'b', 'c', 'd']
for i in range(len(my_list) - 1):
dictionary[my_list[i]] = (my_list[i], ) for i in sorted(dictionary.keys()):
k = dictionary[i]
# Insert your code here.
```

A. print(k['0']) print(k["0"]) print(k) print(k[0])
B. **Answer: D**
C. **Explanation:**
D. Topics: dictionary list for range() tuple dict.keys() sorted()
Try it yourself:

```
dictionary = {}
my_list = ['a', 'b', 'c', 'd']
# for i in range(len(my_list) - 1):
for i in range(3):
dictionary[my_list[i]] = (my_list[i], )
print(dictionary) #  {'a': ('a',), 'b': ('b',), 'c': ('c',)}
# for i in sorted(dictionary.keys()):
for i in dictionary.keys():
k = dictionary[i]
# print(k) #  ('a',) ('b',) ('c',)
# print(k['0']) # TypeError: tuple indices must be integers or slices, not str
# print(k["0"]) # TypeError: tuple indices must be integers or slices, not str
print(k[0])
```

Explanation:
You need k[0] because every element of the dictionary is a tuple and you want the first index of each of those tuples.

Question: 101

What is the expected output of the following code?
```
a =  [1,  2,  3,  4,  5]
print(a[3:0:-1])
```

A. [4, 3, 2,1]
B. [4, 3]
C. [4, 3, 2]
D. The code is erroneous.

Answer: C

Explanation:
Topic: list slicing
Try it yourself:

```
a =  [1, 2, 3, 4, 5]
print(a[3:0:-1])   #  [4, 3, 2]
```

Explanation:
List slicing: [start(inclusive):end(exclusive):step]
The step is negative here.
That makes the slice be build from right to left (start interchanges with end).
The start is the index 3 (inclusive) and the end is index 0 (exclusive).
Therefore we will get index 3 index 2 and index 1
which are the numbers 4, 3, 2

Question: 102

Which of the following variable names are illegal and will cause the SyntaxError exception? (Select two answers)

A. In
B. in
C. print
D. for

Answer: B,D

Explanation:
Topic: variable names
Try it yourself:

```
#  in = 42 # SyntaxError: invalid syntax
#  for = 7 # SyntaxError: invalid syntax
In =  23
print =  42
print(print) # TypeError: 'int' object is not callable
```

Explanation:
in and for are both Python keywords and therefore cannot be used as variable names.

Question: 103

You develop a Python application for your company.
A list named employees contains 200 employee names, the last five being company management.
You need to slice the list to display all employees excluding management. Which code segments can you use?
Choose two.

A. employees[1:-5]
B. employees[0:-4]
C. employees[:-5]
D. employees[0:-5]
E. employees[1:-4]

Answer: C,D **Explanation:**

Topic: list slicing
Try it yourself:

```
employees =  []
# for i in range(1, 196): for i in range(1, 6):       # Just for convenience
employees.append('Employee' + str(i)) for i in range(1, 6): employees.append('Manager' +
str(i)) print(employees) print(employees[:-5]) print(employees[0:-5]) print(employees[1:-4])
-  One manager present and one employee is missing print(employees[1:-5]) # One employee is
missing print(employees[0:-4]) # One manager present
```

Explanation:
List slicing: [start(inclusive):end(exclusive)]
The default value for the start is 0
Meaning you can write it or leave it out.
The negative index counts from the end.
The end is exclusive.
- 1 cuts out one element.
- 5 cuts out five elements.

And that is what you need here for the five managers.

Question: 104

What is the expected output of the following code?

```
x =    1    +    1    //    2    +
   1  / 2 +  2
print(x)
```

A. 3.5
B. 3
C. 4
D. 4.0

Answer: A

Explanation:

Topics: arithmetic operators operator precedence
Try it yourself:

```
x =  1  +1 // 2 + 1 / 2+  2
print(x)                          #  3.5
print(1  + 1  //2 + 1 /  2  +     2) #    3.5
```

```
print(1 +  (1 // 2) +  1   /  2 +   2) #    3.5
print(1 +  0+  1 /2  + 2)          # 3.5
print(1 +  0  +(1 / 2) +  2)       #  3.5
print(1 +  0+  0.5+  2)            # 3.5
print((1   +0)  +0.5  + 2)         # 3.5
print(1 +  0.5 + 2)                # 3.5
print((1 +  0.5) +  2)             #  3.5
print(1.5  +  2)                   # 3.5
print(3.5)                         # 3.5
```

Explanation:
The operators here are from two different groups:
The group "Multiplication, Division, Floor division, Modulus" has a higher precedence than the group "Addition, Subtraction".
Therefore the order of operations here is: // -> / -> + -> + -> +

Question: 105

Which of the following function headers is correct?

A. def func(a=1, b):
B. def func(a=1, b=1, c=2, d):
C. def func(a=1, b=1, c=2):
D. def func(a=1, b, c=2):

Answer: C

Explanation:
Topics: def default arguments
Try it yourself:

```
def func(a=1, b=1, c=2):
pass
#   deffunc(a=1, b): pass           # SyntaxError: ...
#   deffunc(a=1, b, c=2):pass       # SyntaxError: ...
#   deffunc(a=1, b=1, c=2, d):  pass # SyntaxError: ...
#  This would also work:
def func(a, b=1,  c=2):
pass
"""
def func(a=1, b): pass
func(7)
The 7 would go into a and there is nothing left for b.
```

Explanation:
The default argument(s) have to be at the end.

Question: 106

What is the expected output of the following code?

```
def func(message, num=1): print(message * num) func('Hello') func('Welcome', 3)
```

A.

```
Hello
Welcome Welcome Welcome
```

B.

```
Hello
Viewers
```

C.

```
Hello
```

D.

```
Hello
WelcomeWelcomeWelcome
```

E.

```
Hello
Welcome,Welcome,Welcome
```

Answer: D

Explanation:

Topics: def default parameter multiply operator string concatenation

Try it yourself:

```
def func(message, num=1): print(message * num) func('Hello')       # Hello
func('Welcome', 3)  # WelcomeWelcomeWelcome
```

Explanation:

In the function a string concatenation by multiplication takes place.

Once with the default value of num (1)

and once with the one passed by argument (3)

Question: 107

What is the expected output of the following code?

```
def fun(n): n **= n return n print(fun(3))
```

A. The programm will cause an error

B. 27

C. True

D. 9

E. 3

Answer: B

Explanation:

Topics: functions return exponentiation/power operator

Try it yourself:

```
def fun(n): # n **= n n = n ** n return n
print(fun(3)) #   27
```

Explanation:

The fun() function is called with the integer 3

3 to the power of 3 is 27

which is then returned and printed.

Question: 108

What is the expected output of the following code?

```
num = 1
def func():
num = 3
print(num, end=' ')
func()
print(num)
```

A. The code is erroneous.
B. 1 1
C. 3 3
D. 1 3
E. 3 1

Answer: E

Explanation:

Topics: scope def print() with end parameter

Try it yourself:

```
num = 1
def func():
num = 3                # Shadows name 'num' from outer scope
print(num, end=' ')    # 3
func()
print(num) # 1
```

Explanation:

The num variable inside the function scope will be a different variable. It will shadow the name of the num variable from the outer scope, but it will be a different entity.

Question: 109

What is the output of the following snippet?

```
my_list = [x * x for x in range(5)]
def fun(lst):
del lst[lst[2]]
return lst
print(fun(my_list))
```

A. [0, 1, 4,9]
B. [0, 1, 4,16]
C. [0, 1, 9,16]
D. [1, 4, 9,16]

Answer: A

Explanation:

Topics: list comprehension range() multiplication operator functions return del list indexing

Try it yourself:

```
my_list = [x * x for x in range(5)] print(my_list) # [0,    1,    4,    9,    16]
#  The same without list comprehension: my_list =    []
for x in range(5):
my_list.append(x * x)
```

```
print(my_list) #   [0, 1, 4, 9, 16]
def fun(lst):
#  del lst[lst[2]] del lst[4] return lst
print(fun(my_list)) #   [0, 1, 4, 9]
```

Explanation:
The value of index 2 is 4
and therefore index 4 is deleted.

Question: 110

A complete set of known commands is called:

A. an instruction list
B. a machine list
C. a low-level list

Answer: A

Explanation:
Topic: instruction list
Explanation:
A complete set of known commands is called an instruction list, sometimes abbreviated to IL.
Different types of computers may vary depending on the size of their ILs, and the instructions could be completely different in different models.
The IL is, in fact, the alphabet of a machine language.
This is the simplest and most primary set of symbols
we can use to give commands to a computer.
It's the computer's mother tongue.
https://en.wikipedia.org/wiki/Instruction_list

Question: 111

Consider the following code.

```
x =  1
x = x == x
```

The value eventually assigned to x is equal to:

A. False
B. 1
C. True
D. 0

Answer: C

Explanation:
Topic: equal to operator
Try it yourself:

```
x =  1
x = x == x
print(x) # True
x =  1
x = x == x
x =  (1 == 1)
x = True
```

Explanation:
x has the value 1
x gets compared with the equal to operator with itself.
It is true, that x is equal to x
True gets stored in the same variable, in x

Question: 112

What is the expected output of the following code?

```
data =  [
[1, 2,  3, 4],
[5,  6,  7, 8],
[9,  10,  11,  12],
[13,  14,  15, 16]
]


for i in range(0, 4): print(data[i].pop(), end=' ')
```

A. 1 2 3 4
B. 1 5 9 13
C. 13 14 15 16
D. 4 8 12 16

Answer: D

Explanation:
Topics: list for range() pop() print() with end parameter
Try it yourself:

```
data =  [
[1, 2,  3, 4],
[5,  6,  7, 8],
[9,  10,  11,  12],
[13,  14,  15, 16]
]
for i in range(0, 4):
print(data[i].pop(), end=' ')    # 4 8 12 16
print()
print(list(range(0, 4)))   #  [0, 1, 2, 3]
print([1,  2,  3,  4].pop())       # 4
print([5,  6,  7,  8].pop())       # 8
print([9,  10,  11,  12].pop())    # 12
print([13,  14,  15, 16].pop())    # 16
```

Explanation:
data is a two dimensional list.
In every iteration of the for loop,
pop() removes the last element of the corresponding inner list.
That last element gets printed.
The default value of the print() functions end parameter is the line feed.
That gets overwritten here by a single space character.

Question: 113

Consider the following code.

```
for n in range(1,  6,  1):
print(??? *  5)
```

What would you insert instead of ???
so that the program prints the following pattern to the monitor?
11111
22222
33333
44444
55555

A. n
B. 2
C. 1
D. str(n)
E. -1

Answer: D

Explanation:

Topics: for range() str() multiply operator string concatenation
Try it yourself:

```
for n in range(1, 6, 1):
print(str(n) * 5)
,,,,,,
11111
22222
33333
44444
55555 ......
print(' -------  ')
for n in range(1, 6, 1):
print(n *  5)
...
5
10
15
20
25 ...
print(' -------  ')
for n in range(1, 6, 1):
print(-1 *  5)
...
-5
-5
-5
-5
-5 ...
print('  ------  ')
for n in range(1, 6, 1):
print(1 *  5)
...
5
5
5
5
5
print(' -------  ')
for n in range(1, 6, 1):
print(2 *  5)

10
10
10
10
10
```

Explanation:
range(1, 6, 1) delivers the right numbers 1, 2, 3, 4, 5.
(It would also works without step 1, because that's the default value.)
You need the str() function here to get more numbers.
Otherwise a calculation takes place
and you end up with one number per row.
By string concatenation you get the right result:

```
'1' * 5 -> '11111'
'2' * 5 -> '22222'
'3' * 5 -> '33333'
'4' * 5 -> '44444'
'5' * 5 -> '55555'
```

Question: 114

What is the expected output of the following code? `print(chr(ord('z') - 2))`

A. y B. a C. z D. x

Answer: D

Explanation:

Topics: chr() ord()
Try it yourself:

```
print(chr(ord('z') - 2))  # x
print(ord('z'))           # 122
print(chr(120))           # x
```

Explanation:
ord() returns an integer representing the Unicode character.
chr() turns that integer back to the Unicode character.
You don't need to remember the number of every character, but like in the alphabet x is two before z

Question: 115

What is the expected output of the following code?

```
x = """
"""
print(len(x))
```

A. 1
B. The code is erroneous.
C. 0
D. 2

Answer: A

Explanation:

Topics: multiline string len()
Try it yourself:

```
x = """
"""
print(len(x)) #  1
```

```
# ord() returns an integer representing the Unicode character. print(ord(x[0])) #      10
(LF: line feed, new line)
# Same result with single quotes: y =    '''
'''
print(len(y)) #   1
# Every line feed is a character: z =    """
"""
print(len(z)) #   2
```

Explanation:
In a multiline string the line feed gets saved like any other character.

Question: 116

Which of the following lines properly starts a function using two parameters, both with zeroed default values?

A. fun fun(a, b=0):
B. def fun(a=b=0):
C. fun fun(a=0, b):
D. def fun(a=0, b=0):

Answer: D

Explanation:
Topics: functions positional parameters keyword parameters
Try it yourself:

```
def fun(a=0, b=0):
print(a, b)
fun() #   0 0
def fun(a, b=0):
print(a, b)
fun() # TypeError: fun() missing 1 required positional argument: 'a' # def fun(a=0, b):
print(a, b)
# SyntaxError: non-default argument follows default argument
# def fun(a=b=0): print(a, b)
# SyntaxError: invalid syntax
```

Explanation:
There is no multiple assignment inside of the parameter list.
You have to assign each parameter its default value.

Question: 117

What is the output of the following program if the user enters kangaroo at the first prompt and 0 at the second prompt?

```
try: first_prompt = input("Enter the first value: ") a = len(first_prompt)
second_prompt = input("Enter the second value: ") b = len(second_prompt) *      2
print(a/b)
except ZeroDivisionError:
print("Do not divide by zero!")
except ValueError:
print("Wrong value.")
except:
print("Error.Error.Error.")
```

A. Error.Error.Error.
B. Do not divide by zero!
C. 4.0
D. Wrong value.

Answer: C

Explanation:

Topics: try except ZeroDivisionError ValueError input()
len() multiplication operator
Try it yourself:

```
try:
# first_prompt = input("Enter the first value: ") first_prompt = "kangaroo"
a = len(first_prompt)
# second_prompt = input("Enter the second value: ") second_prompt =      "0"
b = len(second_prompt) *  2
print(a/b)                          #  4.0
except ZeroDivisionError:
print("Do not divide by zero!")
except ValueError:
print("Wrong value.")
except:
print("Error.Error.Error.")
```

Explanation:
No exception will be thrown.
The length of kangaroo is 8
and the length of 0 is 1
1 * 2 -> 2
8 / 2 -> 4.0

Question: 118

After execution of the following snippet, the sum of all vals elements will be equal to:

```
vals =  [0, 1, 2]
vals.insert(0, 1) del vals[1]
```

A. 2
8. 4
C. 5
D. 3

Answer: B

Explanation:

Topics: list indexing list.insert() del
Try it yourself:

```
vals =  [0, 1, 2]
vals.insert(0, 1)
print(vals)  # [1, 0, 1, 2]
del vals[1]
print(vals)  # [1, 1, 2]
print(sum(vals)) #   4
```

Explanation:
First 1 is inserted at index 0 and the list is: [1, 0, 1, 2]
Then the new index 1 is deleted and the list is: [1, 1, 2] 1 + 1 + 2 -> 4

Question: 119

What is the expected output of the following code?

```
data =  [1, 2, 3, 4, 5, 6]
for i in range(1, 6):
data[i -  1] = data[i]
for i in range(0, 6): print(data[i], end=' ')
```

A. 1 12 34 5
B. 2 34 56 1
C. 1 23 45 6
D. 2 34 56 6

Answer: D

Explanation:

Topics: for range() print() with end parameter list indexing
Try it yourself:

```
data =  [1, 2, 3, 4, 5, 6]
for i in range(1, 6):  #  1 -> 2 -> 3 -> 4 -> 5
data[i -  1] = data[i]
print(data) #   [2, 3, 4, 5, 6, 6]
# This is just for output:
for i in range(0, 6):
print(data[i], end=' ')  #  2 3 4 5 6 6
```

Explanation:
In the first for loop, the five last values of data get written to the first five indexes of data The value of the last index stays the same. range(1, 6) will produce the numbers 1, 2, 3, 4, 5
And because the first of these number is 1 we need the data[i -1] to start at index 0 The old value at index 1 is 2
That gets written at index 0
And so forth ...

Question: 120

What is the best definition of a script?

A. It's an error message generated by the interpreter
B. It's a text file that contains instructions which make up a Python program
C. It's an error message generated by the compiler
D. It's a text file that contains sequences of zeroes and ones

Answer: B

Explanation:

Topic: script
Explanation:
Due to historical reasons, languages designed for use in interpretation are often called scripting languages, while the source programs encoded using them are called scripts.

Question: 121

What is the expected output of the following code?

```
numbers =  [1, 2, 3, 4, 5]
nums = numbers[2: ]
print(nums)
```

A. [2]
B. [2, 3, 4, 5]
C. The program will cause an error
D. [3, 4, 5]

Answer: D

Explanation:

Topic: list slicing

Try it yourself:

```
numbers =  [1, 2, 3, 4, 5]
nums = numbers[2:]
print(nums) #   [3, 4, 5]
```

Explanation:
[start:stop]
The stop is not given which means the last element is included.
The start is index 2 (inclusive) and therefore the result is [3, 4, 5]

Question: 122

Which of the following statements is false?

A. The result of the / operator is always an integer value.
B. The ** operator has right-to-left associativity.
C. The right argument of the % operator can not be zero.
D. Multiplication precedes addition.

Answer: A

Explanation:

Topic: division operator integer float

Try it yourself:

```
#  "The result of the / operator is always an integer value."
#  is false:
print(9 /  3)  # 3.0
#  The result of the division operator is always a float value
#  "Multiplication precedes addition." is true:
print(3  +4 * 5)    #  23
print(3  +(4 * 5))  #  23
print(3  +20)       #  23
print(23)           #  23
#  "The ** operator has right-to-left associativity." is true:
print(2  ** 3 **  2)   # 512
print(2  ** (3**  2)) # 512
print(2  ** 9)        # 512
print(512)            # 512
#  "The right argument of the % operator cannot be zero."
#  is true:
#  print(7 % 0)   # ZeroDivisionError: ...
```

Explanation:
As written above,
the result of the division operator is always a float value.
Even if it operates with two integer values.

Question: 123

What is the output of the following code?

```
try:
value = input("Enter a value: ") print(value/value)
except ValueError:
print("Bad input...")
except ZeroDivisionError:
print("Very bad input...") except TypeError:
print("Very very bad input...")
except:
print("Booo!")
```

A. Very bad input...
B. Bad input...
C. Booo!
D. Very very bad input...

Answer: D

Explanation:

Topics: try except input() ValueError
TypeError ZeroDivisionError
Try it yourself:

```
try:
# value = input("Enter a value: ")
value =  "100"
print(value/value)
except ValueError:
print("Bad input...")
except ZeroDivisionError:
print("Very bad input...")
except TypeError:
print("Very very bad input...")
E.  Very very bad input...
except:
print("Booo!")
```

Explanation:
The input() function always returns a string.
If you try to divide by a string you get a TypeError

Question: 124

What is the expected output of the following code? `print(1 //  2  *  3)`

A. 0.16666666666666666
B. 0
C. 0.0
D. 4.5

Answer: B

Explanation:

Topics: arithmetic operators operator precedence
Try it yourself:

```
print(1  //2 *  3)    # 0
```

```
print((1  //2)  * 3)  # 0
print(0  *3)          # 0
print(0)              # 0
```

Explanation:
There are only operators from the group
"Multiplication, Division, Floor division, Modulus". The group has a left-to-right associativity, meaning the left one gets evaluated first.
Therefore the order of operations here is: // -> *

Question: 125

Which of the following methods can be used to add an element to a list in Python?

A. .update()
B. .insert()
C. .append()
D. .add()

Answer: C

Explanation:
Correct answer: .append()

Question: 126

What is the purpose of a conditional statement in programming?

A. To execute a block of code based on a condition
B. To iterate over a loop
C. To declare a variable
D. To define a function

Answer: A

Explanation:
Correct answer:
To execute a block of code based on a condition

Question: 127

What is the purpose of the "else" keyword in a conditional statement?

A. To exit the loop and continue with the next iteration
B. To specify an alternative set of code to execute if the condition in the "if" statement is false
C. To define a loop that repeats until a certain condition is met
D. To specify a condition that must be met before executing the code inside the block

Answer: B

Explanation:
(Correct)To specify an alternative set of code to execute if the condition in the "if" statement is false

Question: 128

Which keyword is used to raise exceptions in Python?

A. catch
B. throw
C. handle
D. raise

Answer: D

Explanation:
Correct answer: raise

Question: 129

How do you handle exceptions in Python?

A. By using the "catch" keyword followed by the exception name
B. By using the "try" and "except" keywords
C. By using the "handle" keyword followed by the exception name
D. By using the "throw" keyword followed by the exception name

Answer: B

Explanation:
Correct answer:
By using the "try" and "except" keywords

Question: 130

What is the purpose of the try-except block in Python?

A. To define a new function in Python.
B. To terminate the execution of a program.
C. To handle and catch exceptions that may occur in the code.
D. To execute a block of code only if a certain condition is met.

Answer: C

Explanation:
Correct answer:
To handle and catch exceptions that may occur in the code.

Question: 131

What is the correct syntax to define a tuple in Python?

A. my_tuple = "1, 2, 3" B. my_tuple = (1, 2, 3) C. my_tuple = {1, 2, 3} D. my_tuple = [1, 2, 3] **Answer: B**

Explanation:
Correct answer: my_tuple = (1, 2, 3)

Question: 132

Which data collection in Python uses key-value pairs?

A. List
B. Tuple
C. Dictionary
D. String

Answer: C

Explanation:
Correct answer: Dictionary

Question: 133

What is the main difference between a list and a tuple in Python?

A. Tuples are indexed starting from 0, while lists are indexed starting from 1
B. Lists can store different data types, while tuples can only store one data type
C. Lists are mutable while tuples are immutable
D. All of the above

Answer: C

Explanation:

Correct answer:
Lists are mutable while tuples are immutable

Question: 134

Which data collection in Python is immutable and represented by using single or double quotation marks?

A. Strings
B. Lists
C. Dictionaries
D. Tuples

Answer: A

Explanation:

Correct answer: Strings

Question: 135

What does the "continue" keyword do in a loop?

A. Restarts the loop from the beginning
B. Terminates the loop and continues with the next iteration
C. Exits the loop completely and continues with the next statement outside the loop
D. Skips the current iteration and moves to the next one

Answer: D

Explanation:

Correct answer:
Skips the current iteration and moves to the next one

Question: 136

What is the difference between a compiler and an interpreter?

A. A compiler and an interpreter perform the same function in different programming languages.
B. A compiler translates code into machine language, while an interpreter executes code line by line.
C. A compiler executes code line by line, while an interpreter translates code into machine language. D. A compiler is used for high-level languages, while an interpreter is used for low-level languages.

Answer: B

Explanation:

Correct answer:
A compiler translates code into machine language, while an interpreter executes code line by line.

Question: 137

Which of the following is an example of a built-in exception in Python?

A. IndexOutOfRangeException
B. FileNotFoundException
C. ValueError
D. NullPointerException

Answer: C

Explanation:

Correct answer:
ValueError

Question: 138

Which keyword is used to define a conditional statement in Python?

A. while
B. for
C. if
D. else

Answer: C

Explanation:

Correct answer: if

Question: 139

What does the else statement do in Python?

A. It defines a function and handles exceptions.
B. It executes a block of code when a condition is false.
C. It executes a block of code when a condition is true.
D. It repeats a block of code multiple times.

Answer: B

Explanation:

Correct answer:
It executes a block of code when a condition is false.

Question: 140

Which of the following operations can be performed on a list in Python?

A. Deleting an element by its value
B. Inserting an element at a specific index
C. All of the above
D. Updating the value of an element at a specific index

Answer: C

Explanation:

Correct answer: All of the above

Question: 141

What is the purpose of the return statement in a function?

A. To handle exceptions in the function
B. To print the output of the function
C. To assign a value to a variable within the function

D. To terminate the function and return a value to the caller

Answer: D

Explanation:

Correct answer:

To terminate the function and return a value to the caller

Question: 142

What is the syntax for an if statement in Python?

A. ifcondition:

B. ifcondition =>

C. ifcondition {

D. if condition then:

Answer: A

Explanation:

Correct answer: if condition:

Question: 143

How can you access individual characters in a string in Python?

A. Using the `get()` method

B. Using the `split()` function

C. Using indexing notation

D. Using the `len()` function

Answer: C

Explanation:

Correct answer:

Using indexing notation

Question: 144

Which of the following is the correct syntax for a "for" loop in Python?

A. for i in [1, 2, 3, 4, 5]:

B. for i = 1 to 10:

C. for i in range(10):

D. for i in 1..10:

Answer: C

Explanation:

Correct answer: for i in range(10):

Question: 145

What is the purpose of the "finally" block in a try-except-finally statement?

A. To define a new function in Python.

B. To execute a block of code regardless of whether an exception is raised or not.

C. To handle and catch exceptions that may occur in the code.
D. To execute a block of code only if a certain condition is met.

Answer: B

Explanation:

Correct answer:
To execute a block of code regardless of whether an exception is raised or not.

Question: 146

What is the purpose of the continue statement in Python?

A. It skips the remaining statements in the loop and moves to the next iteration.
B. It repeats a block of code multiple times.
C. It executes a block of code when a condition is false.
D. It defines a function and handles exceptions.

Answer: A

Explanation:

Correct answer:
It skips the remaining statements in the loop and moves to the next iteration.

Question: 147

Which of the following data collections is ordered and allows duplicate values?

A. Tuple
B. String
C. Dictionary
D. List

Answer: D

Explanation:

Correct answer: List

Question: 148

How many times will the code inside a while loop execute?

A. It is not possible todetermine.
B. It depends onthecondition.
C. At least once.
D. Exactly once.

Answer: C

Explanation:

Correct answer:
At least once.

Question: 149

What is the purpose of the break statement in Python?

A. It executes a block of code when a condition is false.
B. It defines a function and handles exceptions.
C. It repeats a block of code multiple times.
D. It skips the remaining statements in the loop and moves to the next iteration.

Answer: D

Explanation:

Correct answer:

It skips the remaining statements in the loop and moves to the next iteration.

Question: 150

What is the output of the following code?

```
num =  10
while num >= 0:
    num -=  2
    print(num)
```

A. 10 9 8 7 6 5 4 3 2 1 0
B. 10 8 6 4 2 0
C. 9 7 5 3 1
D. 0 2 4 6 8 10

Answer: B

Explanation:

Correct answer: 10 8 6 4 2 0

Question: 151

Which of the following data structures is used to store a sequence of characters?

A. String
B. Tuple
C. List
D. Dictionary

Answer: A

Explanation:

Correct answer: String

Question: 152

What is the purpose of the finally block in Python exception handling?

A. To execute a block of code regardless of whether an exception occurred or not
B. To define custom exceptions in the code
C. To skip the remaining code in the try block and move to the next statement
D. To handle exceptions that occur in the try block

Answer: A

Explanation:

Correct answer:

To execute a block of code regardless of whether an exception occurred or not

Question: 153

Which of the following is an example of a mutable data type in Python?

A. Lists

B. Dictionaries
C. Strings
D. Tuples

Answer: A

Explanation:

Correct answer: Lists

Question: 154

Which data type in Python is denoted by enclosing elements in single quotes (') or double quotes ("")?

A. Strings
B. Lists
C. Dictionaries
D. Tuples

Answer: A

Explanation:

Correct answer: Strings

Question: 155

What is the correct syntax to declare a function in Python?

A. functionName():
B. functionName{}
C. functionName[]
D. def functionName:

Answer: D

Explanation:

Correct answer- def functionName:

Question: 156

What is the correct syntax to access a value in a dictionary in Python?

A. dictionary.get(key)
B. dictionary(key)
C. dictionary[key]
D. dictionary.value(key)

Answer: C

Explanation:

Correct answer: dictionary[key]

Question: 157

What is a tuple in Python?

A. Acollectionof elements that areunordered and unchangeable
B. Acollectionof elements that areordered and unchangeable
C. Acollectionof elements that areunordered and changeable
D. Acollectionof elements that areordered and changeable

Answer: B

Explanation:
Correct answer:
A collection of elements that are ordered and unchangeable

Question: 158

Which of the following is NOT a type of exception handling in Python?

A. try-finally
B. try-else
C. try-except
D. try-catch

Answer: D

Explanation:
Correct answer: try-catch

Question: 159

What does the BREAK keyword do in a loop?

A. Stops the execution of the loop and continues with the next statement
B. Terminates the current iteration and resumes the next iteration
C. Skips a certain number of iterations in the loop
D. Restarts the loop from the beginning

Answer: A

Explanation:
Correct answer:
Stops the execution of the loop and continues with the next statement

Question: 160

Which data type in Python is unordered, mutable, and uses key-value pairs to store data?

A. Tuples
B. Strings
C. Dictionaries
D. Lists

Answer: C

Explanation:
Correct answer: Dictionaries

Question: 161

Which of the following data types is ordered, mutable, and allows duplicate elements?

A. Dictionaries
C. Strings
C. Lists
D. Tuples

Answer: C

Explanation:
Correct answer: Lists

Question: 162

Which data type in Python is denoted by enclosing elements in parentheses ()?
A. Tuples
B. Dictionaries
C. Strings
D. Lists
Answer: A
Explanation:
Correct answer: Tuples

Question: 163

What does the CONTINUE keyword do in a loop?

A. Restarts the loop from the beginning
B. Skips a certain number of iterations in the loop
C. Terminates the current iteration and resumes the next iteration
D. Stops the execution of the loop and continues with the next statement
Answer: C
Explanation:
Correct answer:
Terminates the current iteration and resumes the next iteration

Question: 164

Which data type in Python is denoted by enclosing elements in square brackets []?

A. Dictionaries
B. Tuples
C. Lists
E. Strings
Answer: C
Explanation:
Correct answer: Lists

Question: 165

Which of the following data structures is immutable and ordered?

A. Dictionary
B. List
C. String
D. Tuple
Answer: D
Explanation:
Correct answer: Tuple

Question: 166

What is the output of the following Python code?

```
x = "Hello, World!"
```

```
result = x.split(",")
print(result)
```

A. ["Hello", "World!"]
B. ["Hello", " World!"]
C. ["Hello, World!"]
D. "Hello, World!"

Answer: B **Explanation:** Correct answer:
["Hello", " World!"]

Question: 167

What is the purpose of a conditional block in Python?

A. To define a function
B. To execute a block of code when a certain condition is true
C. To iterate over a collection of elements
D. To handle exceptions

Answer: B
Explanation:
Correct answer:
To execute a block of code when a certain condition is true

Question: 168

What is the output of the following Python code? `x = [1, 2, 3, 4, 5]`

```
result = len(x)
print(result)
```

A. [0, 1, 2, 3, 4]
B. [1, 2, 3, 4, 5]
C. 5
D. 15

Answer: C
Explanation:
Correct answer: 5

Question: 169

What will be the output of the following Python code? `x = -5`

```
result = abs(x)
print(result)
```

A. -5
B. 5

C. 0
D. TRUE
Answer: B
Explanation:
Correct answer: 5

Question: 170

Which operator is used to check if two values are equal in a conditional statement?

A. >
B. ==
C. !=
D. <=
Answer: B
Explanation:
Correct answer: ==

Question: 171

What is an exception in Python?

A. An error that occurs during script execution.
B. A value stored in a variable.
C. A condition that results in an endless loop.
D. A specific outcome that triggers a function.
Answer: A
Explanation:
Correct answer:
An error that occurs during script execution.

Question: 172

What is the output of the following Python code?

```
x = "Hello, World!" result = x.replace("o", "a") print(result)
```

A. "Hella,Warld!
B. "Hello,World!
C. "Hella,Warld!
D. "Hello,Warld!
Answer: C **Explanation:** Correct answer: "Hella, Warld!"

Question: 173

What is the purpose of using parameters in a function?

A. To perform calculations.
B. To pass values into a function.
C. To store data temporarily.
D. To control the flow of execution.
Answer: B
Explanation:

Correct answer:
To pass values into a function.

Question: 174

Which of the following methods can be used to add an element at the end of a list in Python?

A. pop()
B. insert()
C. remove()
D. append()

Answer: D

Explanation:
Correct answer: append()

Question: 175

Which keyword is used to define a function in Python?

A. return
B. for
C. def
D. if

Answer: C

Explanation:
Correct answer: def

Question: 176

What will be the output of the following Python code?

```
x =  15
y =  2
result = x // y print(result)
```

A. 5
B. 7
C. 7.5
D. 6

Answer: B

Explanation:

Correct answer: 7

Question: 177

What is the output of the following Python code?

```
my_list =[1,    2,    3,    4,    5]

index = my_list.index(3)

print(index)
```

5
4
2
3

A.
B.
C.
D.

Answer: C

Explanation:

Correct answer: 2

Question: 178

What will be the output of the following Python code?

```
x = "Hello, World!"

result = x.find("W")

print(result)
```

A. 5
B. 7
C. 2
D. 3

Answer: B

Explanation:

(Correct)7

Question: 179

What is the output of the following Python code?

```
x = 7

y = 2
result = x % y

print(result)
```

2
3
0
1

A.
B.
C.
D.

Answer: D

Explanation:

Correct answer: 1

Question: 180

What is the output of the following Python code?

```
x = 3
y = 4

result = x ** y print(result)
```

A. 79
B. 9
C. 80
D. 81

Answer: D

Explanation:

Correct answer: 81

Question: 181

What is the output of the following Python code?

```
my_list =[1, 2, 3, 4, 5]

my_list.remove(3)
print(my_list)
```

A. [1, 3,4,5]
B. [1, 2,3,4, 5]
C. [1, 2,4, 5]
D. [1, 2, 5]

Answer: C

Explanation:
Correct answer: [1, 2, 4, 5]

Question: 182

What is the output of the following Python code?
```
my_list =[1,    2,    3,    4,    5]
my_list.remove(3)

print(my_list)
```

A. [1, 3,4, 5]
B. [2, 3,4, 5]
C. [1, 2,4, 5]
D. [1, 2, 3, 4, 5]

Answer: C

Explanation:
Correct answer: [1, 2, 4, 5]

Question: 183

What is the output of the following Python code?
```
my_tuple =    (1,    2,    3,    4,    5)

slice_tuple = my_tuple[1:4]
print(slice_tuple)
```

A. (1, 2, 3)
B. (2, 3,4, 5)
C. (2, 3, 4)
D. (1, 2, 3, 4, 5)

Answer: C

Explanation:
Correct answer: (2, 3, 4)

Question: 184

What is the output of the following Python code?
```
my_list =[1,    2,    3,    4,    5]
new_list = my_list.copy()

print(new_list)
```

A. [1, 2, 3, 4, 5, 7]
B. [1, 3, 5]
C. [1, 2, 3,4, 5]
D. [5, 4, 3,2, 1]

Answer: C

Explanation:
Correct answer: [1, 2, 3, 4, 5]

Question: 185

What is the output of the following Python code?

```
my_list =[1,    2,    3,    4,    5]

del my_list[2]

print(my_list)
```

A. [1, 2, 3, 4, 5]
B. [1, 3, 4, 5]
C. [1, 2, 5]
D. [1, 2, 4, 5]

Answer: D

Explanation:

Correct answer: [1, 2, 4, 5]

Question: 186

What will be the output of the following Python code?

```
x =  [1,    2,    3,    4,    5]

result = x[3]

print(result)
```

A. 4
B. 3
C. 5
D. 2

Answer: A

Explanation:

Correct answer: 4

Question: 187

What is the output of the following Python code?

```
my_tuple =      (1,    2,    3)

my_tuple[2] = 4

print(my_tuple)
```

A. [1, 2, 3]
B. (1, 2, 3)
C. Error: 'tuple' object does not support item assignment
D. (1, 2, 4)

Answer: C

Explanation:

Correct answer:
Error: 'tuple' object does not support item assignment

Question: 188

What is the output of the following Python code?

```
my_list =[1,    2,    3,    4,    5]
new_list = sorted(my_list, reverse=True)

print(new_list)
```

A. [1, 2, 5,4, 3]
B. [1, 2, 3,4, 5]
C. [5, 4, 3,2, 1]
D. [1, 3, 2,5, 4]

Answer: C

Explanation:

Correct answer: [5, 4, 3, 2, 1]

Question: 189

What will be the output of the following Python code?

```
x = " Hello "

result = x.strip()
print(result)
```

A. " Hello
B. "Hello"

C. "Hello "
D. " Hello "
Answer: B
Explanation:
Correct answer: "Hello"

Question: 190

What is the output of the following Python code?

```
my_tuple = (1, 2, 3, 4, 5)

max_value = max(my_tuple)
print(max_value)
```

3
2
4
5

A.
B.
C.
D.
Answer: D
Explanation:
Correct answer: 5

Question: 191

What is the output of the following Python code?

```
my_tuple = (1, 2, 3, 6, 4, 5)

length = len(my_tuple)

print(length)
```

6
7
5
4

A.
B.
C.
D.
Answer: A
Explanation:
Correct answer : 6

Question: 192

What is the output of the following Python code?
```
my_tuple = (1, 2, 3, 4, 5)
new_tuple = my_tuple[-3:-1]
print(new_tuple)
```

A. (2, 3)
B. (1, 2)
C. (3, 4)
D. (4, 5)

Answer: C

Explanation:
Correct answer: (3, 4)

Question: 193

What is the output of the following Python code?
```
my_list =[1, 2, 3, 4, 5]
my_list.insert(2, 6)
print(my_list)
```

A. [1, 2,6, 3,4, 5]
B. [1, 2, 3, 4, 5]
C. [1, 2,3, 4,5, 6]
D. [1, 6, 2, 3,4, 5]

Answer: A

Explanation:
Correct answer: [1, 2, 6, 3, 4, 5]

Question: 194

What will be the output of the following Python code?
```
x = 7
y = 3
result = x > y
print(type(result))
```

A. str
B. bool
C. int
D. float

Answer: B

Explanation:
Correct answer: bool

Question: 195

What is output of the following code?
```
x = 4
y = 10

while x > y:
x -= 1

y += 1

else:
y -= 1

print(y)
```

A. 12
B. 9
C. 8
D. 11

Answer: B

Explanation:
Correct answer: 9

Question: 196

What is the output of the following Python code?
```
my_list =[1, 2, 3, 4, 5]

reversed_list = my_list[::-1]

print(reversed_list)
```

A. [1, 2, 3, 4, 5]
B. [5, 3, 1]
C. [1, 3, 5]
D. [5, 4, 3, 2, 1]

Answer: D

Explanation:
Correct answer: [5, 4, 3, 2, 1]

Question: 197

What is the output of the following Python code?
```
my_list =[1, 2, 3, 4, 5]

repeated_list = my_list * 2

print(repeated_list)
```

A. [1, 1, 2, 2, 3, 3, 4, 4, 5, 5]
B. [1, 2, 3, 4, 5, 1, 2, 3, 4, 5]
C. [2, 4, 6, 8, 10]
D. [1, 2, 3, 4, 5, 2, 4, 6, 8, 10]

Answer: B

Explanation:
Correct answer: [1, 2, 3, 4, 5, 1, 2, 3, 4, 5]

Question: 198

What is the output of the following Python code?

```
my_list =[1,    2,    3,    4,    5]
popped_element = my_list.pop(2)

print(popped_element)
```

A. 2
B. 4
C. 3
D. 5
Answer: C
Explanation:
Correct answer: 3

Question: 199

What is output of the following code?

```
x =   5

y =  3
if x > y:

z = x + y

elif x < y:
z = x - y

else:
z = x * y

print(z)
```

A. 8
B. 14
C. 9
D. 10
Answer: A
Explanation:
Correct answer: 8

Question: 200

What is output of the following code?

```
x =   4

y =  8
```

```
for i in range(x, y):
    if i % 2 == 0:
        continue
    else:
        print(i)
```

A. 4, 5,6, 7
B. 4, 5, 6, 7, 8
C. 5, 7
D. 4, 6, 8

Answer: C

Explanation:
Correct answer: 5, 7

Question: 201

What is the output of the following Python code? `my_tuple = (1, 2, 3, 4, 5)`

```
new_tuple = my_tuple[::-1]
print(new_tuple)
```

A. (1, 2, 3, 4, 5)
B. (5, 3, 1)
C. (5, 4, 3)
D. (5, 4, 3, 2, 1)

Answer: D

Explanation:
Correct answer: (5, 4, 3, 2, 1)

Question: 202

Which of the following is not a keyword?

A. nonlocal
B. assert
C. eval
D. pass

Answer: C

Explanation:
Correct answer: eval

Question: 203

Which of the following is an invalid statement?

A. a_b_c = 1,000,000
B. a,b,c = 1000, 2000, 3000
C. abc = 1,000,000

D. a b c = 1000 2000 3000

Answer: D

Explanation:

Correct answer:

a b c = 1000 2000 3000

Question: 204

Which of the following is not a complex number?

A. k= 2+31

B. k =2+ 3j

C. k= 2+3J

D. k = complex(2, 3)

Answer: A

Explanation:

Correct answer: k = 2 + 31

Question: 205

What is the output of the following Python code?

```
x = "Python"

result = x[1:5:2]

print(result)
```

A. Py

B. on

C. en

D. yh

Answer: D Explanation:

Correct answer: yh

Question: 206

What will be the value of the following Python expression? `4 + 3 % 5`

A. 3.0
B. 5
C. 4
D. 7

Answer: D

Explanation:

Correct answer:

The order of precedence is: %, +. Hence the expression above, on simplification results in 4 + 3 = 7. Hence the result is 7.

Question: 207

What is the value of the following expression? `float(22//3+3/3)`

A. "8"
B. "8.0"
C. "8.33"
D. "8.3"

Answer: B

Explanation:

Correct answer: "8.0"

Question: 208

What will be the output of the following Python code?

```
x = 123

for i in x:

print(i)
```

A. error
B. none of the mentioned
C. 1 2 3
D. 123

Answer: A

Explanation:

Correct answer: error

Question: 209

What will be the value of x in the following Python expression? `x =int(43.55+2/2)`

A. 33
B. 22
C. 44
D. 52

Answer: C

Explanation:
Correct answer: 44

Question: 210

What will be the output of the following Python code? `d = {0, 1, 2}`

```
for x in d:
print(x)
```

A. error
C. none of the mentioned
C. 0 1 2
D. {0, 1, 2} {0, 1, 2} {0, 1, 2}

Answer: C

Explanation:
Correct answer: 0 1 2

Question: 211

Which one of these is floor division?

A.
D. //
E. None of the mentioned
F. /
G. %

Answer: A

Explanation:

Correct answer: //

Question: 212

What error occurs when you execute the following Python code snippet?

A. Orange = mango
B. NameError
C. TypeError
D. ValueError
E. SyntaxError

Answer: B

Explanation:

Correct answer: NameError

Question: 213

What is the output of the following Python code?

```
x = "Hello" result = x[::-3] print(result)
```

A. oe
B. Hello olleH elloH
C. **Answer: A**
D. **Explanation:**
Correct answer: oe

Question: 214

What will be the output of the following Python code snippet?

```
x  = 'abcd'
for i in range(len(x)):
x  = 'a' print(x)
```

A. a
B. none ofthementioned
C. abcd abcdabcd
D. a a a a

Answer: D

Explanation:

Correct answer: a a a a

Question: 215

What is the output of the following Python code?

```
x = 7
result = x % 2
print(result)
```

A. 3
B. 0
C. 2
D. 1

Answer: D

Explanation:
Correct answer: 1

Question: 216

Why are local variable names beginning with an underscore discouraged?

A. they confuse the interpreter
B. they are used to indicate global variables
C. they slow down execution
D. they are used to indicate a private variables of a class

Answer: D

Explanation:
Correct answer:
they are used to indicate a private variables of a class

Question: 217

What does ~4 evaluate to?

A. -3
B. -5
C. -4
D. 3

Answer: B

Explanation:
Correct answer: -5

Question: 218

Which of the following is true for variable names in Python?

A. all private members must have leading and trailing underscores B. none of the mentioned

C. unlimited length
D. underscore and ampersand are the only two special characters allowed

Answer: C

Explanation:

Correct answer: unlimited length

Question: 219

Operators with the same precedence are evaluated in which manner?

A. Right to Left
B. None of the mentioned
C. Cana€™t say
D. Left to Right

Answer: D

Explanation:

Correct answer: Left to Right

Question: 220

What is the output of the following Python code?

```
x = "Hello" result = x.replace("l", "X") print(result)
```

A. HeXXo
B. Hello
C. Hexxo
D. XeXXo

Answer: A

Explanation:

Correct answer: HeXXo

Question: 221

What is the output of the following Python code?

```
x = "Hello" result = x[1:4] print(result)
```

A. ello ell llo Hell
B. **Answer: B**
C. **Explanation:**
D. Correct answer: ell

Question: 222

What will be the output of the following Python code?

```
d = {0: 'a', 1: 'b', 2: 'c'}
for x, y in d.items(): print(x, y)
```

A. 0 12
B. none of the mentioned
C. a bc
D. 0 a 1 b 2 c

Answer: D

Explanation:

Correct answer: 0 a 1 b 2 c

Question: 223

Is Python case sensitive when dealing with identifiers?

A. s
B. s

Answer: B

Question: 224

What is the output of the following Python code?

```
x =  5
y =  "2"
result = str(x) + y print(result)
```

A. 7
8. 52
C. 15
D. 5

Answer: B

Explanation:

Correct answer: 52

Question: 225

Please fill the blank field(s) in the statement with the right words. Is Python case sensitive when dealing with identifiers?

A. Yes

Answer: A

Question: 226

What will be the value of the following Python expression? `4+2**5//10`

A. 5
B. 7
C. 77
D. 55

Answer: B

Explanation:

Correct answer: 7

Question: 227

What is the result of cmp(3, 1)?

A. 1
B. FALSE
C. TRUE
D. 0

Answer: A

Explanation:

Correct answer: 1

Question: 228

Which of the following cannot be a variable?

A. it
B. in
C. on
D. init

Answer: B

Explanation:

Correct answer: in

Question: 229

Select all options that print. hello-how-are-you

A. print(a€~hello-a€~ + a€~how-are-youa€™)
B. print(a€~helloa€™ + a€~-a€~ + a€~howa€™ + a€~-a€~ + a€~areа€™ + a€~youa€™)

C. print(a€~helloa€™, a€~howa€™, a€~area€™, a€~youa€™ + a€~-a€~ * 4)
D. print(a€~helloa€™, a€~howa€™, a€~area€™, a€~youa€™)
Answer: A
Explanation:
Correct answer:
print(a€~hello-a€~ + a€~how-are-youa€™)

Question: 230

What is the output of this expression, 3*1**3?

A. 27
B. 1
C. 3
D. 9
Answer: C
Explanation:
Correct answer: 3

Question: 231

Which of the following is incorrect?

A. float(a€™12+34a€²)
B. float(a€~infa€™)
C. float(a€™56a€™+a€™78a€™)
D. float(a€~nana€™)
Answer: A
Explanation:
Correct answer: float(a€™12+34a€2)

Question: 232

What is the return value of trunc()?

A. int
B. None
C. float
D. bool
Answer: A
Explanation:
Correct answer: int

Question: 233

What will be the output of the following Python code?

```
d = {0: 'a', 1: 'b', 2: 'c'}
for x in d.keys():
print(d[x])
```

A. 0 1 2
B. none of the mentioned
C. 0 a 1 b 2 c
D. a b c

Answer: D

Explanation:

Correct answer: a b c

Question: 234

Which of the following is incorrect?

A. x = 19023
B. x = 0x4f5
C. x = 30963
D. x = 03964

Answer: D

Explanation:

Correct answer: x = 03964

Question: 235

Which one of the following has the highest precedence in the expression?

A. Multiplication
B. Addition
C. Exponential
D. Parentheses

Answer: D

Explanation:

Correct answer: Parentheses

Question: 236

What will be the output of the following Python code snippet?

```
x = 2
for i in range(x):
x -= 2
print (x)
```

A. 0 -2
B. 0 1 2 3 4 a€j
C. error
D. 0

Answer: A

Explanation:

Correct answer: 0 -2

Question: 237

What is the output of print (0.1 + 0.2 == 0.3)?

A. FALSE
B. Machine dependent
C. TRUE
D. Error

Answer: A

Explanation:
Correct answer: FALSE

Question: 238

Which of the following results in a SyntaxError?

A. a€reHe said, a€~Yes!'a€^
B. a€~3\a€™
C. a€^€™Thata€™s okaya€^€™
D. a€~a€^Once upon a timea€|a€% she said.a€™

Answer: B

Explanation:
Correct answer: a€~3\a€™

Question: 239

Which of the following is the truncation division operator?

A. %
B. |
C. /
D. //

Answer: D

Explanation:
Correct answer: //

Question: 240

Which of the following operators has its associativity from right to left?

A. +
B. //
**
D. %

Answer: B **Explanation:**
Correct answer:

Question: 241

What will be the output of the following Python code?

```
d = {0, 1, 2}
for x in d.values(): print(x)
```

A. 0 1 2
B. error
C. none of the mentioned
D. None

Answer: B

Explanation:
Correct answer: error

Question: 242

What will be the value of X in the following Python expression?

```
X = 2+9*((3*12)-8)/10
```

A. 25
B. 22
C. 27.2
D. 27.6

Answer: C

Explanation:
Correct answer: 27.2

Question: 243

Evaluate the expression given below if A = 16 and B = 15. A % B // A

A. "0"
B. "1.0"
C. 1
D. "0.0"

Answer: A

Explanation:
Correct answer: "0"

Question: 244

What is the value of the following expression? `2+4.00, 2**4.0`

A. (6.00, 16.0)
B. (6.0, 16.0)
C. (6, 16)
D. (6.00, 16.00)

Answer: B

Explanation:
Correct answer: (6.0, 16.0)

Question: 245

Which of the following will run without errors?

A. round(6352.898,2,5)
B. round()
C. round(7463.123,2,1)
D. round(45.8)

Answer: D

Explanation:
Correct answer: round(45.8)

Question: 246

What are the values of the following Python expressions? 2**(3**2) (2**3)**2 2**3**2

A. 512, 512, 512
B. 64, 64, 64
C. 512, 64, 512
D. 64, 512, 64

Answer: C

Explanation:
Correct answer: 512, 64, 512

Question: 247

Which of the following represents the bitwise XOR operator?

A. ^
B. &
C. |
D. !

Answer: A

Explanation:
Correct answer: ^

Question: 248

What will be the output of the following Python expression? `24//6%3, 24//4//2`

A. (1,3)
B. (3,1)
C. (1,0)
D. (0,3)

Answer: A

Explanation:
Correct answer: (1,3)

Question: 249

Please fill the blank field(s) in the statement with the right words.

A. Bitwise gives 1 if either of the bits is 1 and 0 when both of the bits are 1.
B. XOR

Answer: B

Question: 250

Which one of the following has the same precedence level?

A. Addition and Multiplication
B. Multiplication, Division, Addition and Subtraction
C. Multiplication, Division and Addition
D. Addition and Subtraction

Answer: D

Explanation:

Correct answer: Addition and Subtraction

Question: 251

Which of the following is a Python tuple?

A. (1, 2, 3)
B. {1, 2, 3}
C. [1, 2, 3]
D. {}

Answer: A

Explanation:

Correct answer: (1, 2, 3)

Question: 252

To which of the following the â€œinâ€• operator can be used to check if an item is in it?

A. Dictionary
B. All of the mentioned
C. Lists
D. Set

Answer: B

Explanation:

Correct answer: All of the mentioned

Question: 253

Suppose list1 is [2445,133,12454,123], what is max(list1)?

A. 133
B. 2445
C. 123
D. 12454

Answer: D

Explanation:

Correct answer: 12454

Question: 254

Please fill the blank field(s) in the statement with the right words.
Program code making use of a given module is called a of the module.

A. Client

Answer: A

Question: 255

If a={5,6,7,8}, which of the following statements is false?

A. a[2]=45
B. print(min(a))
C. print(len(a))
D. a.remove(5)

Answer: A

Explanation:

Correct answer: a[2]=45

Question: 256

To concatenate two strings to a third what statements are applicable?

A. s3 = s1.add(s2)
B. s3 = s1. add (s2)

C. s3 =s1 * s2
D. s3 =s1 . S2
Answer: B
Explanation:
Correct answer: s3 = s1. add (s2)

Question: 257

Which of these about a frozenset is not true?

A. Data type with unordered values
B. Allows duplicate values
C. Mutable data type
D. Immutable data type
Answer: C
Explanation:
Correct answer: Mutable data type

Question: 258

Which of the following statements is used to create an empty set?

A. set()
B. ( )
C. { }
D. [ ]
Answer: A
Explanation:
Correct answer: set()

Question: 259

Which of the following commands will create a list?

A. all of the mentioned
B. list1 = list()
C. list1 = []
D. list1 = list([1,2,3])
Answer: A
Explanation:
Correct answer:
all of the mentioned

Question: 260

Please fill the blank field(s) in the statement with the right words. Set members must not be hashable .

A. False
B. Explanation
C. False
Answer: A

Question: 261

What will be the output of the following Python code?

```
>>>  a=(0,1,2,3,4)
>>> b=slice(0,2)
>>> a[b]
```

A. (0,2)
B. (0,1)
C. Invalid syntax for slicing
D. [0,2]

Answer: B

Explanation:

Correct answer: (0,1)

Question: 262

To insert 5 to the third position in list1, we use which command?

A. list1.insert(3, 5)
B. list1.add(3, 5)
C. list1.append(3, 5)
D. list1.insert(2, 5)

Answer: D

Explanation:

Correct answer: list1.insert(2, 5)

Question: 263

Which of the following statements create a dictionary?

A. d= {40:â€•johnâ€•, 45:â€•peterâ€•}
B. d = {â€œjohnâ€•:40,â€œpeterâ€•:45}
C. All of the mentioned
D. d = {}

Answer: C

Explanation:

Correct answer: All of the mentioned

Question: 264

Please fill the blank field(s) in the statement with the right words.

A. If a=(1,2,3,4), a[1:-1] is
B. (2,3)

Answer: B

Question: 265

What is called when a function is defined inside a class?

A. Method
B. Class
C. Module
D. Another function

Answer: A

Explanation:
Correct answer: Method

Question: 266

To remove string â€œhelloâ€• from list1, we use which command?

A. list1.remove(â€œhelloâ€•)
B. list1.remove(hello)
C. list1.removeOne(â€œhelloâ€•)
D. list1.removeAll(â€œhelloâ€•)

Answer: A

Explanation:
Correct answer: list1.remove(â€œhelloâ€•)

Question: 267

What arithmetic operators cannot be used with strings?

A. "+"
B. "-"
C. "*"
D. All of the mentioned

Answer: B

Explanation:
Correct answer: "-"

Question: 268

Please fill the blank field(s) in the statement with the right words.

A. Which is the correct operator for power(x y)?
B. **

Answer: B

Question: 269

What will be the output of the following Python code?

```
l1=[10, 20, 30]
l2=[-10, -20, -30]
l3=[x+y for x, y in zip(l1, 12)]
print(l3)
```

A. Error
B. [-20, -60, -80]
C. 0
D. [0, 0, 0]
Answer: D
Explanation:
Correct answer: [0, 0, 0]

Question: 270

Please fill the blank field(s) in the statement with the right words. Set makes use of Dictionary makes use of

A. Keys, Key Values
Answer: A

Question: 271

What type of data is: a=[(1,1),(2,4),(3,9)]?

A. Array of tuples
B. List of tuples
C. Tuples of lists
D. Invalid type
Answer: B
Explanation:
Correct answer: List of tuples

Question: 272

Which of these about a dictionary is false?

A. Dictionaries arenâ€™t ordered
B. The keys of a dictionary can be accessed using values
C. The values of a dictionary can be accessed using keys
D. Dictionaries are mutable

Answer: B

Explanation:

Correct answer:
The keys of a dictionary can be accessed using values

Question: 273

To add a new element to a list we use which command?

A. list1.addEnd(5)
B. list1.add(5)
C. list1.append(5)
D. list1.addLast(5)

Answer: C

Explanation:

Correct answer: list1.append(5)

Question: 274

What is the data type of (1)?

A. Both tuple and integer
B. Tuple
C. List
D. Integer

Answer: D

Explanation:

Correct answer: Integer

Question: 275

A nested list is given:

```
stocks = [['AAPL', 'MSFT'], ['AMZN', 'TSLA', 'NVDA']]
```

Which of the following code snippets makes a deep copy of the stocks list? (select 1)

A.

```
1.  import copy
2.
3.  stocks_copy = copy.deepcopy(stocks)
```

B.

```
1. import copy
2.
3. stocks_copy = copy.copy(stocks)
```

C.

```
stocks_copy  = stocks.copy()
```

D.

```
stocks_copy  = list.copy(stocks)
```

Answer: A

Explanation:

The difference between shallow and deep copying is only relevant for compound objects (objects that contain other objects, like lists or class instances):

A shallow copy constructs a new compound object and then (to the extent possible) inserts references into it to the objects found in the original.

A deep copy constructs a new compound object and then, recursively, inserts copies into it of the objects found in the original.

Question: 276

What is the result of the following code snippet?

```
numbers = list(range(4))
for i in range(3):
numbers.insert(-1, numbers[i]) print(numbers)
```

A. [0,1,2,3, 0,1,2, 3, 4]
B. [0,1,2,3, 0,1,2]
C. [0,1,2,0, 1,2,3]
D. [0,1,0,1, 2,2]

Answer: C

Explanation:

list.insert(i, x) - insert an item at a given position. The first argument is the index of the element before which to insert, so a.insert(0, x) inserts at the front of the list, and a.insert(len(a), x) is equivalent to a.append(x).

https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 277

The following function is given:

```
def add(a, b): result = a + b return
```

What the function returns with the following call:

A. add(3, 5)
B. '35'
C. None
D. 35
E. 8

Answer: C

Explanation:

This function returns None because of the use of return keyword.

Question: 278

What built-in function can you use to display the documentation for a function, class, etc.

A. info() B. help() C. docs() D. dir() **Answer: B**

Explanation:
help([object]) - invoke the built-in help system.
https://docs.python.org/3/library/functions.html#help

Question: 279

What is a recursive function?

A. A recursive function is a function that can only be called once.
B. A recursive function is a function that calls itself infinitely many times.
C. A recursive function is any function implemented using lamba expression.
D. A recursive function is a function that calls itself one or more times initsbody.

Answer: D

Question: 280

Which of the following code snippets shows the correct way to handle multiple exceptions in a single except clause?

A.
```
except: TypeError, ValueError # your code
```

B.
```
except (TypeError, ValueError) # your code
```

C.
```
except: (TypeError, ValueError)
```

```
# your code
```

D.

```
except (TypeError, ValueError): # your code
```

E.

```
except TypeError, ValueError # your code
```

Answer: D

Explanation:

An except clause may name multiple exceptions as a parenthesized tuple, for example: except (RuntimeError, TypeError, NameError):
pass

Question: 281

What built-in function can you use to convert integer to binary?

A. bin() float() bool() hex() chr() int()
B. **Answer: A**
C. **Explanation:**
D. bin(x) - convert an integer number to a binary string prefixed with a€reGba€'.
E. https://docs.python.org/3/library/functions.html#bin
F.

Question: 282

What is the result of the following code snippet?

```
numbers = list(range(10))
numbers = numbers[::-1]
numbers = numbers[::2]
print(numbers)
```

A. [0, 2, 4, 6, 8]
B. [10, 8, 6, 4, 2, 0]
C. [9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
D. [9, 7, 5, 3, 1]

Answer: D

Explanation:

List slicing: list[start:stop:step]

Question: 283

How can you assign number 1,000,000 (int type) to the variable in Python? (select 2)

A. var = 1e6
B. var = 1_000_000
C. var = 1.000.000
D. var = 1000000
E. var = 1,000,000

Answer: B,D

Explanation:

var = 1_000_000

Python 3.6 introduced underscores in numeric literals, which allows you to place single underscores between numbers and after base specifiers to improve readability. This feature is not available in older Python versions.

var = 1e6

This is a float type.

var = 1,000,000

It will raise the SyntaxError.

var = 1.000.000

It will raise the SyntaxError.

Question: 284

Who is the creator of Python?

A. Brendan Eich
B. James Gosling
C. Guido van Rossum
D. Dennis Ritchie
E. Linus Torvalds

Answer: C

Explanation:

https://en.wikipedia.org/wiki/Guido_van_Rossum

Question: 285

Python is an example of...

A.... a machine language.
B.... a high-level programminglanguage.
C ... a natural language.
D ... a low-level programminglanguage.

Answer: B

Question: 286

What is the output of the following print function?

A. print('C:\Windows\Users\krako')
B. C: Windows Users krako
C. C:\Windows\Users\krako
D. C:\Windows\Users\krako
E. C:
Windows
Users
krako

Answer: D

Explanation:

\ is the escape character in Python.

Question: 287

The following lists are given:
letters = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H']
numbers = [1, 2, 3, 4, 5, 6, 7, 8]
What does the list below look like?

A. chessboard = [(i, j) for i, j in zip(letters, numbers)]
B. ['A1B2C3D4E5F6G7H8']
C. [('A', '1'), ('B', '2'), ('C', '3'), ('D', '4'), ('E', '5'), ('F', '6'), ('G', '7'), ('H', '8')]
D. ['A1', 'B2', 'C3', 'D4', 'E5', 'F6', 'G7', 'H8']
E. ['A1 B2 C3 D4 E5 F6 G7 H8']
F. [('A', 1), ('B', 2), ('C', 3), ('D', 4), ('E', 5), ('F', 6), ('G', 7), ('H', 8)]

Answer: F

Explanation:
zip(*iterables, strict=False) - Iterate over several iterables in parallel, producing tuples with an item from each one.
https://docs.python.org/3/library/functions.html#zip

Question: 288

The following operations were performed:

```
a = [1, 2, 3, 4, 5]
b = a
b[:] = [3, 6]
```

Select the correct answer.

A. a = [1, 2, 3, 4, 5] b = [1, 2, 3, 4, 5]
B. a = [1, 2, 3, 4, 5] b = [3, 6]
C. a = [3, 6]
b = [3, 6]
D. a = [1, 2, 3, 4, 5]
b = [1, 2, 3, 4, 5, 3, 6]

Answer: C

Explanation:
a and b are the same list.

Question: 289

What is the result of the following code snippet?

```
stocks = ['Apple', 'Samsung', 'HP'] while len(stocks):
print(stocks.pop())
```

A. Apple Samsung HP
B. HP Samsung Apple
C. HP

D. Apple

Answer: B

Explanation:

list.pop([i]) - remove the item at the given position in the list, and return it. If no index is specified removes and returns the last item in the list.

len(s) - return the length (the number of items) of an object.

Question: 290

We have two sets:

```
junior = {'python', 'html', 'css'}
senior = {'python', 'html', 'css', 'django', 'javascript'}
```

How do you check if the junior set is a subset of the senior set? (select 2)

A. junior | senior
B. junior <= senior
C. junior.issubset(senior)
D. junior < senior

Answer: B,C

Explanation:

issubset(other) or set <= other - test whether every element in the set is in other.

https://docs.python.org/3/library/stdtypes.html#frozenset.issubset

Question: 291

This function returns the next element from the iterator. If the iterator is exhausted, a StopIteration exception will be raised. You can also pass a default value to this function, which will be returned instead of raising a StopIteration exception.

A. Which built-in function does this description apply to?
B. iter()
C. next()
D. func()
E. gen()

Answer: C

Explanation:

next(iterator[, default]) - retrieve the next item from the iterator by calling its next () method. If default is given, it is returned if the iterator is exhausted, otherwise StopIteration is raised.

https://docs.python.org/3/library/functions.html#next

Question: 292

What is the LEGB rule for?

A. This is the rule by which Python manages memory.
B. This is the rule by which Python handles exceptions.
C. This is the rule by which Python executes instructions.
D. This is the rule by which Python resolves names. The letters in LEGB stand for Local, Enclosing,

Global and Built-in scope.
Answer: D

Question: 293

What is a Python docstring and what is it for?

A. Docstring is a string that appears at the beginning of a module, function, class, or method definition. The docstring passed in this way becomes a special doc attribute for this object. For consistency, use """triple quotes""" around the docstring.
B. Docstring is a Python-built class for creating text documents.
C. Docstring is a string that appears at the beginning of a module, function, class, or method definition. The docstring passed in this way becomes a special name attribute for this object. For consistency, use """triple quotes""" around the docstring.

Answer: A

Explanation:
A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. Such a docstring becomes the doc special attribute of that object.
https://peps.python.org/pep-0257/#what-is-a-docstring

Question: 294

The following dictionary is given:
companies = {'TSLA': 800.0, 'NVDA': 145.6}
What value will be returned when trying to read 'AMZN' key?

A. False
B. None
C. The NameError exception will be raised.
D. 800.0
E. The IndexError exception will be raised.
F. The KeyError exception will be raised.

Answer: F

Explanation:
KeyError is raised when a mapping (dictionary) key is not found in the set of existing keys.
https://docs.python.org/3/library/exceptions.html#KeyError

Question: 295

What is the value assigned to the var variable?

```
var = 10 % 3
var = var == 1
```

A. 1
B. True
C. False
D. 3

Answer: B

Explanation:
Step 1: var = 10 % 3 var = 1
Step 2: var = var == 1 var = 1 == 1 var = True

Question: 296

Which of the following words is not a Python keyword? (select 3)

A. go
B. loop
C. yield
D. finally
E. switch
F. lambda
G. break

Answer: A,B,E

Explanation:
List of Python keywords:

```
[
    'False', 'None', 'True',
    'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del',
    'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in',
    'is', 'lambda', 'nonlocal', 'not',
    'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield',
]
```

https://docs.python.org/3/reference/lexical_analysis.html#keywords

Question: 297

What is the result of the following code snippet?

```
translator = {}
translator['1'] = (1,'one')
translator['2'] = (2,'two')
translator['3'] = (3,'three'

for item in translator.keys(): print(translator[item][1])
```

This code will raise the IndexError exception.

A. 1
2
3
B. one
two
three
C. (1, 'one')
(2, 'two')
(3, 'three')

Answer: B

Explanation:
translator list:
translator = {'1': (1, 'one'), '2': (2, 'two'), '3': (3, 'three')}

Question: 298

What is a global variable?

A. Aglobal variable is any variable defined inside a class.
B. Aglobal variable is a variable that is available in the globalscope.
C. Aglobal variable is a variable that is available in the enclosingscope.
D. Aglobal variable is any variable defined inside a function.

Answer: B

Explanation:
global_stmt ::= "global" identifier ("," identifier)*
The global statement is a declaration which holds for the entire current code block. It means that the listed identifiers are to be interpreted as globals. It would be impossible to assign to a global variable without global, although free variables may refer to globals without being declared global.
https://docs.python.org/3/reference/simple_stmts.html#the-global-statement

Question: 299

Select which variable names are correct. (select 3)

A. del
B. s300
C. compute_mean
D. 100_
E. Logistic Regression
F. LogisticRegression
G. 303

Answer: B,C,F

Explanation:
del is a Python keyword.
100_, 303 - variable name cannot start with a digit
Logistic Regression - space character is not allowed in a variable name

Question: 300

The following dictionaries are given:

```
stocks_1 = {'MSFT': 280.0, 'AAPL': 146.39, 'TSLA': 644.22}
stocks_2 = {'AMZN': 3573.0, 'NVDA': 726.44}
```

We want to combine them into one. Select correct answers. (select 3)

A. stocks 1 + stocks 2
B. Python 3.9.0 or higher: stocks_1 | stocks_2
C. Python 3.5.0 or higher: {**stocks_1, **stocks_2}
D. stocks_1.update(stocks_2)

Answer: B,C,D

Explanation:
dict.update([other]) - updates the dictionary with the key/value pairs from other, overwriting existing keys. Return None.
https://docs.python.org/3/library/stdtypes.html#dict.update

Question: 301

What is the expected output of the following code?

```
tech_name = 'Python 3.10'

for character in tech_name:
     if character.isupper():
          print(character.lower(), end='')
     elif character.islower():
          print(character.upper(), end='')
     elif character.isspace() or not character.isdigit(): continue
     else:
          print(character)
```

A. pYTHON3.10
B. PYTHON310
C. pYTHON310
D. pYTHON 3.10

Answer: C

Question: 302

What is the result of the following code snippet? `numbers = [i +  2 for i in range(5)]`

```
def preprocess(input_list):
     del input_list[input_list[1]]
     return input_list
print(preprocess(numbers))
```

A. [2, 4, 5,6]
B. [2, 3, 4,6]
C. [2, 3, 4,5, 6]
D. [2, 3, 4,5]

Answer: B

Explanation:

Steps:
numbers -> [2, 3, 4, 5, 6]
del input_list[input_list[1]] -> del input_list[3]

Question: 303

What is the result of the following code snippet if the user enters 13 and 2 respectively?

```
a = int(input('Enter the first number: '))
b = int(input('Enter the second number: '))

a  = a / b
b  = b // a

print(f'a =  {a}, b =  {b}')
```

A. a = 0.0 b = 6.5

B. This code will raisethe SyntaxError exception.
C. This code will raisethe TypeErrorexception.
D. a = 6.5, b = 0.0
E. a = 6.0, b = 2.0

Answer: D

Explanation:
a = a / b -> 13 / 2 -> 6.5
b = b // a -> 2 // 13 -> 0.0

Question: 304

What's wrong with the following code?

```
user_input = input("Enter a number: ") result = user_input **      2.0
print(f'{user_input} to the power of 2 is {result}')
```

A. The ** operator does not exist in Python. The ^ operator is used for exponentiation.
B. The result of the input() function is a string. This code will raise the TypeError exception.
C. Everything is ok.
D. You cannot format code like in the print() function.

Answer: B

Explanation:
Example:
user_input = input("Enter a number: ") -> '10'
result = user_input ** 2.0 -> '10' ** 2.0 -> It will raise the TypeError exception.

Question: 305

You have the following code:

```
print('=' *  50)
print('Welcome! Type \'help()\' to display the documentation.') print('=' *     50)
```

Which returns the following output:
Welcome! Type 'help()' to display the documentation.

How to modify this code to get the following result? (select 2)

Welcome!
Type 'help()' to display the documentation.

About my program: * version: 1.0 * author: krako * language: Python

A.

```
print('=' *  50)
print('Welcome!
B.    Type \'help()\' to display the documentation.')
print('=' *  50)
print('About my program:')
print(' \ t* version: 1.0\  n')
print('   \t* author: krako\ n')
print('   \t* language: Python\ n')
```

```
print('=' *  50)
```

C.

```
print('=' * 50)
print('Welcome! \ tType \'help()\' to display the documentation.')
print('=' * 50)
print('About my program:')
print('\ n* version: 1.0  \ t\ n* author: krako \ t\ n* language: Python')
print('=' *  50)
```

D.

```
print('=' *  50)
print('Welcome!
E.    Type \'help()\' to display the documentation.')
print('=' *  50)
print('About my program:')
print(' \ t* version: 1.0\ n \ t* author: krako\ n \ t* language: Python')
print('=' *  50)
```

F.

```
print('=' * 50)
print('Welcome!\ nType \'help()\' to display the documentation.') print('='      *      50)
print('About my program:')
print('   \ t* version: 1.0')
print('   \ t* author: krako')
print(' \ t* language: Python')
print('=' *  50)
```

Answer: D,F

Question: 306

What is the easiest way to remove duplicates from the list?

A. Use the list.remove_duplicates() method.
B. Convert alist toa set and back to a list.
C. Convert alist toa tuple and back to a list.
D. Convert alist toa dictionary and back to a list.

Answer: B

Question: 307

What will be the result of the following operation? list() is list()

A. None
B. True False NoneType
C. **Answer: C**
D. **Explanation:**
False because these two lists are two different objects.

Question: 308

What instruction will you use to catch exceptions?

A. assert
B. catch
C. raise
D. throw
E. try... except...

Answer: E

Explanation:
The try statement specifies exception handlers and/or cleanup code for a group of statements: try_stmt ::= try1_stmt | try2_stmt
try1_stmt ::= "try" ":" suite
("except" [expression ["as" identifier]] ":" suite)+
["else" ":" suite]
["finally" ":" suite]
try2_stmt ::= "try" ":" suite
"finally" ":" suite
https://docs.python.org/3/tutorial/errors.html#handling-exceptions

Question: 309

Select correct answers about the break statement. (select 2)

A. If the break statement is in a nested loop (a loop inside another loop), the break statement will end the innermost loop.
B. The break statement terminates the loop that contains it. Program control flows to the statements immediately after the loop.

C. The break statement doesn't exist in Python.
D. The break statement is used to skip the rest of the code inside the loop for the current iteration only. The loop does not end but continues with the next iteration.

Answer: A,B

Explanation:

The break statement, like in C, breaks out of the innermost enclosing for or while loop.
https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops

Question: 310

If we iterate over a dictionary, by default what elements is it iterated over?

A. values
B. keys
C. (key: value) pairs
D. (value: key) pairs

Answer: B

Explanation:

Example:

```
stocks = {'Apple': 180, 'Facebook': 40}
for stock in stocks:
print(stock)
for stock in stocks.keys():
print(stock)
for stock in stocks.items():
print(stock)
for stock in stocks.values():
print(stock)
```

Output:

```
Apple
Facebook
Apple
Facebook
('Apple', 180)
('Facebook', 40)
180
40
```

Question: 311

What is the result of the following code snippet?

```
items = list(('ball', 'pen', 'glass', 'book', 'pen', 'ball')) items = set(items)
items = sorted(items)
print(items)
```

A. Thiscode will output ['ball', 'ball', 'book', 'glass', 'pen', 'pen'] to theconsole.
B. Thiscode will output ['ball', 'book', 'glass', 'pen'] to the console.
C. Thiscode will output {'ball', 'book', 'glass', 'pen'} to the console.
D. Thiscode will output {'book', 'ball', 'glass', 'pen'} to the console.
E. Thiscode will output ['ball', 'pen', 'glass', 'book', 'pen', 'ball'] to theconsole.

Answer: B

Explanation:
sorted(iterable, /, *, key=None, reverse=False) - returns a new sorted list from the items in iterable.
https://docs.python.org/3/library/functions.html#sorted

Question: 312

Which of the following operators is not a logical operator?

A. not
B. %
C. and
D. or

Answer: B

Explanation:
The % (modulo) operator yields the remainder from the division of the first argument by the second.

Question: 313

The following stocks list is given:
stocks = ['AAPL', 'MSFT', 'NVDA', 'UBER', 'TSLA']
Select code that will remove 'UBER' from stocks list. Select all that apply.
Expected output:

A. ['AAPL', 'MSFT', 'NVDA', 'TSLA']
B. list.remove(stocks, 'UBER')
C. del stocks[3]
D. del stocks[stocks.index('UBER')]
E. stocks.remove('UBER')

Answer: B,C,D,E

Explanation:
list.remove(x) - remove the first item from the list whose value is equal to x. It raises a ValueError if there is no such item.
https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 314

Select all true statements about tuples. (select 2)

A. Tuples cannot contain other tuples.
B. Each tuple element may be of a different type (i.e., integers, strings, booleans, etc.)
C. You can append elements to tuples.
D. Tuples are ordered and unchangeable (immutable) collections of data.

Answer: B,D

Explanation:
Tuples can contain other tuples.
You cannot append elements to tuples.

Question: 315

Which line properly invokes the function defined as below? (select 2) def function(x, y, z=0):

pass

A. function(0)
B. function(x=1, y=0, z=0)
C. function(1, y=1)
D. function(y=0, y=1)

Answer: B,C

Explanation:
function(0) -> function missing 1 required positional argument: y
function(y=0, y=1) -> keyword argument repeated

Question: 316

Which list method finds the index of the first searched item in the list, or raises a ValueError when no such item is in the list?

A. list.remove()
B. list.index()
C. list.count()
D. list.find()
E. list.pop()

Answer: B

Explanation:
list.index(x[, start[, end]]) - return zero-based index in the list of the first item whose value is equal to x. Raises a ValueError if there is no such item. The optional arguments start and end are interpreted as in the slice notation and are used to limit the search to a particular subsequence of the list. The returned index is computed relative to the beginning of the full sequence rather than the start argument.

Question: 317

What is the result of the following code snippet?

```
code = 've-nf-sv' for char in code: if char == '-':
continue
print(char, end='')
```

A. ve-
B. venfsv
C. ve
D. ve-nf-sv

Answer: B

Explanation:
The continue statement:
continue_stmt ::= "continue"
continue may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop. It continues with the next cycle of the nearest enclosing loop.
https://docs.python.org/3/reference/simple_stmts.html#continue

Question: 318

The following list is given:
numbers = list(range(10, 100, 10))

What will be the result of the following code?
numbers[::-2]

A. [10, 20]
B. [30, 40, 50,60,70, 80, 90]
C. [90, 80, 70,60,50, 40, 30, 20, 10]
D. [90, 70, 50,30,10]

Answer: D

Explanation:
The following reverse list:
[IN]: numbers = list(range(10, 100, 10))
[IN]: numbers[::-1]
[OUT]: [90, 80, 70, 60, 50, 40, 30, 20, 10]
The following reverse list and return every second element:
[IN]: numbers = list(range(10, 100, 10))
[IN]: numbers[::-2]
[OUT]: [90, 70, 50, 30, 10]

Question: 319

The following two lists are given:
stocks_1 = ['TSLA', 'MSFT', 'AMZN']
stocks_2 = ['AAPL', 'NVDA']
How do you combine these lists into one (see below)? (select 3)
Expected output:

A. ['TSLA', 'MSFT', 'AMZN', 'AAPL', 'NVDA']
B. stocks_1 + stocks_2
C. [**stocks_1, **stocks_2]
D. [*stocks_1, *stocks_2]
E. stocks_1.extend(stocks_2)

Answer: B,D,E

Question: 320

You have the following code:

```
user_input = input("Enter a number: ") result = user_input **      2.0
print(f'{user_input} to the power of 2 is {result}')
```

User enters 10. The result of the input() function is a string. How can you modify this code to get rid of the TypeError exception and get the result 100.0? (select 2)

A.

```
user_input = bool(input("Enter a number: ")) result = user_input **      2.0
print(f'{user_input} to the power of 2 is {result}')
```

B.

```
user_input = int(input("Enter a number: ")) result = user_input ** 2.0
print(f'{user_input} to the power of 2 is {result}')
```

C.

```
user_input = complex(input("Enter a number: ")) result = user_input **      2.0
print(f'{user_input} to the power of 2 is {result}')
```

D.

```
user_input = float(input("Enter a number: ")) result = user_input **      2.0
print(f'{user_input} to the power of 2 is {result}')
```

Answer: B,D

Explanation:

You can use int() or float() function to convert str to int or float.

Question: 321

The following list is given:
artists = ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
What does the list below look like?

A. result = [artist[:3] for artist in artists]
B. ['ng', 'l Collins', ' Police', 'en', 'DC']
C. ['Sti', 'Phi', 'The', 'Que', 'AC/']
D. [True, True, True, True, True]
E. ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']

Answer: C

Explanation:

List comprehension basic syntax: [expression for value in iterable]

Question: 322

What exception will be raised when you refer to a variable that has not yet been defined?

A. NameError
B. IndexError
C. KeyError
D. SyntaxError

Answer: A

Explanation:

NameError is raised when a local or global name is not found. This applies only to unqualified names. The associated value is an error message that includes the name that could not be found.
https://docs.python.org/3/library/exceptions.html#NameError

Question: 323

What is the expected behavior of the following program?

```
i = 1
sum = 0
while True:
sum += i
i += 1
print(sum)
```

A. This code will enter an infinite loop.
B. This code will output 0 to the console.
C. This code will not make a single pass through the loop.
D. This code will output 1 to the console.

Answer: A

Explanation:

dict does not have a method like get_values().

Question: 324

What is the result of the following code snippet?

```
coding = {1:2, 2:3, 3:1}
val = coding[2]
for key in range(len(coding)):
val = coding[val]
print(val)
```

A. (1, 2, 3)
B. 1
C. 2
D. 3

Answer: D

Explanation:

The range type represents an immutable sequence of numbers and is commonly used for looping a

specific number of times in for loops.
https://docs.python.org/3/library/stdtypes.html#ranges

Question: 325

What is the result of the following code snippet if the user enters 13 and 2 respectively?

```
a = input('Enter the b first number: ')
= input('Enter the a = second number: ')
a / b
b = b // a
print(f'a =  {a}, b = {b}')
a =  0.0 b = 6.5
```

A. This code will raise the TypeError exception.
B. a = 6.0, b = 2.0
C. This code will raise the SyntaxError exception.
D. a = 6.5, b = 0.0

Answer: A

Explanation:
a = a / b -> '12' / '2'
b = b // a -> '2' // '13'

Question: 326

What is the output of the following snippet?
print(True > False, True < False, True == False)

A. 1 0 0
B. True False False
C. None None None
D. Boolean values cannot be compared with each other and the TypeError will be raised.
E. False True False

Answer: B

Question: 327

Which of the following values is of int type? (select 2)

A. 1e6
B. 10
C. 10.0
D. 10.
E. 0

Answer: B,E

Explanation:
10.0, 10. and 1e6 are float values.

Question: 328

What is the result of the following code snippet?

```
def calculate(a): if a ==  10:
return a
```

```
else:
return calculate(a - 1)
print(calculate(8))
```

A. This code will raise the SyntaxError exception.
B. This code will output 8 to the console.
C. This code will output 10 to the console.
D. This code will raise the RecursionError exception.

Answer: D

Explanation:
calculate(8) -> calculate(7) -> calculate(6) -> calculate(5) -> calculate(4) -> ...

Question: 329

What is a set in Python?

A. Aset isan unordered collection of objects that canbeduplicated.
B. Aset isan ordered collection of unique objects.
C. Aset isan ordered collection of objects that can beduplicated.
D. Aset isan unordered collection of unique objects.

Answer: D

Explanation:
A set is an unordered collection with no duplicate elements.
https://docs.python.org/3/tutorial/datastructures.html#sets

Question: 330

You only need to print the \ (backslash) character to the console. How to do it in Python?

A. print('/\')
B. print('\')
C. print('\\')
D. print('\')

Answer: B

Explanation:
\ is an escape character in Python.

Question: 331

Select all true statements about dictionaries in Python. (select 2)
A. Dictionaries are immutable.
B. Each dictionary is a set of key: value pairs.
C. A key may be any immutable type of object. It can be a number (int or float), or even a string,
D. A list can be a key in a dictionary.

Answer: B,C

Explanation:
A list can be a key in a dictionary. -> False. Lists are mutable.
Dictionaries are immutable. -> False. Dictionaries are mutable.

Question: 332

What built-in function can you use to do modulo division?

A. divint()
B. modulo()
C. divmod()
D. div()

Answer: C

Explanation:
divmod(a, b) - take two (non-complex) numbers as arguments and return a pair of numbers consisting of their quotient and remainder when using integer division. With mixed operand types, the rules for binary arithmetic operators apply. For integers, the result is the same as (a // b, a % b).
https://docs.python.org/3/library/functions.html#divmod

Question: 333

Select the correct answer about the continue statement. (select 2)

A. The continue statement terminates the loop that contains it. Program control flows to the statements immediately after the loop body.
B. There is no continue statement in Python.
C. continue may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop.
D. The continue statement is used to skip the rest of the code inside the loop for the current iteration only. The loop does not end but continues with the next iteration.

Answer: C,D

Explanation:
continue_stmt ::= "continue"
continue may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop. It continues with the next cycle of the nearest enclosing loop.
https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops
https://docs.python.org/3/reference/simple_stmts.html#the-continue-statement

Question: 334

What is the result of the following code snippet? data_stream = ''
while data_stream:
print('data received')
else:
print('data not received')

A. This code will output data not received to the console.
B. This code will output data received to the console.
C. This code will raise SyntaxError exception. You cannot use else clause with while loop.
D. This code will go into an infinite loop.

Answer: A

Explanation:
The while statement is used for repeated execution as long as an expression is true: while_stmt ::= "while" assignment_expression ":" suite ["else" ":" suite]
This repeatedly tests the expression and, if it is true, executes the first suite; if the expression is false (which may be the first time it is tested) the suite of the else clause, if present, is executed and the loop terminates.
https://docs.python.org/3/reference/compound_stmts.html#the-while-statement

Question: 335

Select all true statements about conditional statement in Python. (select 2)

A. The else part is optional in the conditional statement.
B. The following conditional statement is correct: print('some value'): if 'data'
C. There can be at most one elif part in a conditional statement.
D. The keyword elif is short for 'else if'.

Answer: A,D

Explanation:
The if statement is used for conditional execution:
if_stmt ::= "if" assignment_expression ":" suite
("elif" assignment_expression ":" suite)*
["else" ":" suite]
It selects exactly one of the suites by evaluating the expressions one by one until one is found to be true; then that suite is executed (and no other part of the if statement is executed or evaluated). If all expressions are false, the suite of the else clause, if present, is executed.
https://docs.python.org/3/tutorial/controlflow.html#if-statements

Question: 336

How to declare two variables in one line? (select 2)

A. tech_name = 'Python'; version = '3.9.0'
B. tech_name = 'Python': version = '3.9.0'
C. tech_name = 'Python', version = '3.9.0'
D. tech_name, version = 'Python', '3.9.0'

Answer: A,D

Question: 337

The following variable is defined: height = 40
What will happen when executing the following code:

A. assert height > 0
B. The assert statement checks if the value of the height variable is greater than 0. If the condition is true (and in this case it is - it returns True), an AssertionError exception will be raised.
C. The assert statement checks if the value of the height variable is greater than 0. If the condition is false (and in this case it is - it returns False), no error will be raised and code execution continues on the next line (if any).
D. The assert statement checks that the value of the height variable is greater than 0. If the condition is

true (and in this case it is - it returns True), no error will be raised and code execution continues on the next line (if any).

Answer: D

Explanation:

Assert statements are a convenient way to insert debugging assertions into a program: assert_stmt ::= "assert" expression ["," expression]
https://docs.python.org/3/reference/simple_stmts.html#the-assert-statement

Question: 338

Can a tuple be a key in a dictionary?

A. No.
B. Yes, as long as it consists of immutable objects.

Answer: B

Explanation:

Example:
{('Apple', 'AAPL'): 182.5, ('Facebook', 'FB'): 190.0}
But this is incorrect (list - mutable type):
{('Apple', ['APPL', 'INC']): 182.5, ('Facebook', ['FB', 'INC']): 190.0}

Question: 339

What does PSF stand for?

A. Python Software Foundation
B. Programming Software for Python
C. Package Software Format
D. Python Security Fundamentals

Answer: A

Explanation:

https://www.python.org/psf/

Question: 340

What is the result of the following code snippet?

```
i = 10
while True and i > 0:
if i % 2 == 0:
print(i) i -=3
```

A.

```
10
8
6
4
2
10
```

B. This code will enter an infinite loop.
C. 10
4

Answer: C

Explanation:

The while statement is used for repeated execution as long as an expression is true: while_stmt ::= "while" assignment_expression ":" suite ["else" ":" suite]

This repeatedly tests the expression and, if it is true, executes the first suite; if the expression is false (which may be the first time it is tested) the suite of the else clause, if present, is executed and the loop terminates.

https://docs.python.org/3/reference/compound_stmts.html#the-while-statement

Question: 341

Suppose you have the following snippet of code:

```
stocks = [
"Apple",
"Tesla",
"Amazon",
"Microsoft",
"Netflix",
"Alphabet", ]
tmp = stocks
```

Select all true statements. (select 2)

A. stocks has the same length as tmp.
B. stocks and tmp are different list.
C. stocks and tmp are different names of the same list.
D. stocks is a list and tmp is a tuple.

Answer: A,C

Question: 342

What is the result of the following code snippet?
val1 = 10 val2 = 20
val3 = val1 > val2 or val2 % val1 == 0
print(val3)

A. True
B. None
C. False
D. This code will raise the SyntaxError exception.

Answer: A
Explanation:
Steps:
val1 > val2 -> False
val2 % val1 == 0 -> True
val3 = val1 > val2 or val2 % val1 == 0
val3 = False or True
val3 = True

Question: 343

A computer program which directly executes instructions written in a programming language is called...

A. an interpreter.
B. a compiler.
C. a modifier.
D. a translator.

Answer: A

Question: 344

The following list is given: numbers = [0, 1, 2, 3, 4]
What will be the result of the following operation?
sorted(numbers) is numbers

A. NoneType
B. False
C. True
D. None

Answer: B
Explanation:
sorted(iterable, /, *, key=None, reverse=False) - return a new sorted list from the items in iterable.
https://docs.python.org/3/library/functions.html#sorted

Question: 345

Can a function have default argument values?

A. Yes
B. No
Answer: A
Explanation:
Example: def add(a, b=2): return a + b print(add(3))

Question: 346

What is the difference between the list.append() and list.extend() methods?

A. The list.extend() method adds its argument as a single item to the end of the list. The length of the list itself is increased by one. The list.append() method iterates over its argument, adding each item to the list, extending the list. The length of the list will increase by the number of items in the iterable argument.
B. There is no difference between the list.append() and list.extend() methods. These methods can be used interchangeably.
C. The list.append() method adds its argument as a single item to the end of the list. The length of the list itself is increased by one. The list.extend() method iterates over its argument, adding each item to the list, extending the list. The length of the list will increase by the number of items in the iterable argument.
Answer: C **Explanation:** list.append(x) - add an item to the end of the list. Equivalent to a[len(a):] = [x]. list.extend(iterable) - extend the list by appending all the items from the iterable. Equivalent to a[len(a):] = iterable.

Question: 347

What method should you use when iterating over a dictionary to iterate over both the keys and the values of the dictionary?

A. dict.values()
B. dict.iteritems()
C. dict.items()
D. dict.keys()
Answer: C
Explanation:
dict.items() - return a new view of the dictionarya€™s items ((key, value) pairs).
https://docs.python.org/3/library/stdtypes.html#dict.items

Question: 348

What is the result of the following code snippet?

```
var =     50
if var ==  50:
     print(f'var =  {var}')
```

```
if  var % 10 == 0:
     print(f'var %  10 = {var % 10}')
if  var > 10:
     print('var >  10')
else:
     print('end')
```

A. end
B. var = 50
C. var = 50
var % 10 = 0 var > 10 end
E. var = 50 var % 10 = 0 var > 10 **Answer: D**

Question: 349

What is local scope?

A. From the moment we start the program in Python, we are in the local scope. Python turns the main script into a module called main that stores the execution of the main program. The namespace of this module is the local scope of the program.
B. Global scope only appears when you nest functions within other functions.
C. Local scope is created when the function is called. Each time we call the function we create a new local scope. By default, variables inside functions only exist in the local scope associated with the function call. When the function exits, the local scope is destroyed.
Answer: A

Question: 350

What is the result of the following code snippet? numbers = list(range(5)) numbers.insert(0, 1) del numbers[3] print(numbers)

A. [1, 0,1, 2, 3, 4]
B. [0, 1,2, 4, 1]
C. [0, 0,1, 3, 4]
D. [1, 0,1, 3, 4]
Answer: D
Explanation:
Steps:
numbers = list(range(5)) -> [0, 1, 2, 3, 4] numbers.insert(0, 1) -> [1, 0, 1, 2, 3, 4] del numbers[3] -> [1, 0, 1, 3, 4]
list.insert(i, x) - inserts an item at a given position. The first argument is the index of the element before which to insert, so a.insert(0, x) inserts at the front of the list, and a.insert(len(a), x) is equivalent to a.append(x).
https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 351

A function definition...

A. may be placed anywhere inside the code after the first invocation.
B. must be placed before the first invocation.
C. cannot be placed among other code.
D. must be placed after the first invocation.

Answer: B

Question: 352

Suppose you have the following snippet of code: numbers = [1, 2, 3, 4, 5] temp = numbers del temp[:] numbers, temp
Select the true statement.

A. numbers and temp lists have the same length.
B. numbers list is longer than temp list.
C. Both lists have 5 elements.
D. temp list is longer than numbers list.

Answer: A

Explanation:
numbers and temp are the same list.

Question: 353

The following stocks list is given:
stocks = ['AAPL', 'MSFT', 'NVDA', 'UBER', 'TSLA']
Select code that will remove all items from the stocks list. (select 2)
Expected output:

A. []
B. del stocks
C. stocks[::-1]
D. del stocks[:]
E. stocks.clear()

Answer: D,E

Explanation:
list.clear() - remove all items from the list. Equivalent to del a[:].
https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 354

What is the result of the following operation?
'#' * 5
'#5'
The operation cannot be performed. You cannot multiply an object of type str by an object of type int.

A. '55555'
B. '#####'
C. '#'

Answer: B

Question: 355

The following list is given:
stream = ['0343', '5-355', '5452', None]

Is it possible to sort the above list with the list.sort() method?

A. No, because a list that contains None cannot be sorted.
B. No, because the lists contain objects of type str that cannot be sorted.
C. Yes.
Answer: A

Question: 356

Which version of the code is more desirable?

A.
```
x, y =  1, 0
try: x / y
except ValueError as e: print(e)
```
B.
```
x, y =  1, 0
try: x / y except Exception as e: print(e)
```
C.
```
x, y =  1, 0
try: x / y except TypeError as e: print(e)
```
D.
```
x, y =  1, 0
try: x / y
except ZeroDivisionError as e: print(e)
```

Answer: D
Explanation:
The Zen of Python: Explicit is better than implicit.
https://docs.python.org/3/tutorial/errors.html#handling-exceptions

Question: 357

The following list is given:
tech_names = ['python', 'django', 'pandas']
How do you remove all items from the list? (select 2)

A. del tech_names[:]
B. del tech_names
C. rm tech_names
D. tech_names.clear()
Answer: A,D
Explanation:
list.clear() - remove all items from the list. Equivalent to del a[:].

Question: 358

What is the result of the following code snippet?
try:
val = 10 / 0
print(val)
break
except (ValueError, ZeroDivisionError):
print('ValueError or ZeroDivisionError exception raised...')
except:

print('Something went wrong...')

A. This code will raise a ZeroDivisionError exception and output the following message: 'ValueError or ZeroDivisionError exception raised...'
B. The code will have an exception handled by the last except block.
C. This code will raise a ValueError exception and output the following message: 'ValueError or ZeroDivisionError exception raised...'
D. This code will raise a SyntaxError exception (break outside loop).

Answer: D

Question: 359

Consider the function below:

```
def func(number, result=[]): result.append(number) return result
```

We have a classic Python problem here. The default value for the result argument is a list (a mutable object). In Python, the default value of a function argument is only evaluated once.
The function was called twice:
func(10)
func(20)

Considering the above, what will be the result of the next call func(30)?

A. []
B. [20, 30]
C. [10, 20, 30]
D. [30]
E. 30

Answer: C

Explanation:
Pythona€™s default arguments are evaluated once when the function is defined, not each time the function is called.
https://docs.python-guide.org/writing/gotchas/#mutable-default-arguments

Question: 360

The following very_important_fuction() function is given. How do you display the docstring for a given function? (select 3)

```
def very_important_fuction():
     """This is very important function that always returns 1.""" return 1
```

A. bin(very_important_fuction)
B. print(very_important_fuction)
C. (in IPython): very_important_fuction?
D. help(very_important_fuction)
E. (in IPython): ?very_important_fuction

Answer: C,D,E

Explanation:
https://peps.python.org/pep-0257/

Question: 361

Which of the following operators is not a bitwise operator?

A. $
B. ~
C. <<
D. ^
E. |
F. >>
G. &

Answer: A

Explanation:
Bitwise And a & b
Bitwise Exclusive Or a ^ b
Bitwise Inversion
~ a
Bitwise Or a | b
Right Shift a >> b
https://docs.python.org/3/reference/expressions.html#unary-arithmetic-and-bitwise-operations

Question: 362

What built-in function can you use to check if a given object has a specific attribute/method?

A. setattr()
B. getattr()
C. hasattr()
D. attr()

Answer: C

Explanation:
hasattr(object, name) - the arguments are an object and a string. The result is True if the string is the name of one of the objectâ€™s attributes, False if not.
https://docs.python.org/3/library/functions.html#hasattr

Question: 363

Which year is considered the beginning of Python?

A. 2006
B. 1999
C. 1992
D. 1991
E. 2010

Answer: D

Explanation:
Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0.

https://en.wikipedia.org/wiki/Python_(programming_language)

Question: 364

What is the result of the following code snippet?

```
def compute(val):
    if val % 3 == 0:
        return 1
    elif val % 3 == 1:
        return 2
    else:
        return 3

print(compute(compute(compute(5))))
```

A. 3
B. 2
C. 1
D. This code will raise the SyntaxError exception.

Answer: B

Explanation:

Steps:
compute(compute(compute(5))) -> compute(compute(3)) -> compute(1) -> 2

Question: 365

What is the built-in function repr() for?

A. Executes the Python code passed as a string.
B. There is no built-in repr() function in Python.
C. Returns an informal representation of an object as a string.
D. Returns the formal representation of an object as a string.

Answer: D

Explanation:

repr(object) - return a string containing a printable representation of an object. For many types, this function makes an attempt to return a string that would yield an object with the same value when passed to eval(); otherwise, the representation is a string enclosed in angle brackets that contains the name of the type of the object together with additional information often including the name and address of the object. A class can control what this function returns for its instances by defining a repr () method.
https://docs.python.org/3/library/functions.html#repr

Question: 366

The following variable was created:
course_name = 'python interview'
in a global scope. How can you remove this variable? (select 3)

A. remove course_name
B. del course_name
C. rm course_name
D. del globals()['course_name']
E. del(course_name)

Answer: B,D,E

Explanation:

del_stmt ::= "del" target_list
https://docs.python.org/3/reference/simple_stmts.html#the-del-statement

Question: 367

Can a list store objects of different types?

A. No
B. Yes
Answer: B
Explanation:
Example:
items = ['Apple', 30, True, None, (3, 4)]

Question: 368

Let's consider the following problem. A list of numbers is given:
numbers = [1, 2, 3, 4, 5, 6]
Our task is to leave only even numbers in the list. We may be tempted to write the following code:

```
for i in numbers:
     if i %  2 != 0 and i < len(numbers):
          del numbers[i]
```

This solution is incorrect because we iterate over the object from which we remove elements (this causes some element skips in the iterable object and results in a wrong answer).
Select the obtained numbers list.

A. [1, 3, 4, 6, 7, 8]
B. []
C. [3, 4, 6, 7, 8]
D. [1, 3, 4, 6, 7]
Answer: A
Explanation:
Correct implementation (Iterating over the copy of the list):
for i in numbers[:]:
if i % 2 != 0 and i < len(numbers):
del numbers[i]

Question: 369

What does *args mean in function?

A. The special *args syntax is most commonly used in function definitions. It is used to pass a variable number of arguments to a function - arguments with specific names.
B. The special *args syntax is most commonly used in function definitions. It is used to pass a variable number of arguments to a function - arguments that do not have a specific name.
C. Python does not have a *args syntax.
Answer: B

Explanation:
A function can be called with an arbitrary number of arguments. These arguments will be wrapped up in a tuple. Before the variable number of arguments, zero or more normal arguments may occur. Example:

```
def write_multiple_items(file, separator, *args):
file.write(separator.join(args))
```

Normally, these variadic arguments will be last in the list of formal parameters, because they scoop up all remaining input arguments that are passed to the function. Any formal parameters which occur after the *args parameter are â€˜keyword-onlyâ€™ arguments, meaning that they can only be used as keywords rather than positional arguments.

https://docs.python.org/3/tutorial/controlflow.html#arbitrary-argument-lists

Question: 370

What optional clauses does a try ... except ... statement have? (select 2)

A. after before else finally
B. **Answer: C,D**
C. **Explanation:**
D. The try statement specifies exception handlers and/or cleanup code for a group of statements:
try_stmt ::= try1_stmt | try2_stmt try1_stmt ::= "try" ":" suite
("except" [expression ["as" identifier]] ":" suite)+
["else" ":" suite]
["finally" ":" suite]
try2_stmt ::= "try" ":" suite
"finally" ":" suite
https://docs.python.org/3/reference/compound_stmts.html#the-try-statement

Question: 371

What is a Python dictionary?

A. Dictionary is built-in data type in Python. Dictionaries contain pairs of values and corresponding keys, and are indexed by values, which can be of any mutable type.
B. Dictionary is built-in data type in Python. Dictionaries contain pairs of keys and their corresponding values, and are indexed by keys, which can be of any mutable type.
C. Dictionary is built-in data type in Python. Dictionaries contain pairs of keys and their corresponding values, and are indexed by keys, which can be of any immutable type.
D. A dictionary is a collection of all the statements that are available in Python.

Answer: C

Explanation:
Dictionaries are sometimes found in other languages as â€œassociative memoriesâ€• or â€œassociative arraysâ€•. Unlike sequences, which are indexed by a range of numbers, dictionaries are indexed by keys, which can be any immutable type; strings and numbers can always be keys.
https://docs.python.org/3/tutorial/datastructures.html#dictionaries

Question: 372

If you want to give a name to a variable in Python, you must follow some strict rules. Select all true statements. (select 2)

A. The name of the variable must not be any of Python's reserved words.
B. The name of the variable must begin with a digit.
C. Uppercase and lowercase letters are treated as different.
D. The underscore character cannot be used in the variable name.

Answer: A,C

Explanation:
The name of the variable must begin with a letter.
The underscore character can be used in the variable name.

Question: 373

What built-in function can you use to check the logical value of an object?

A. id()
B. bool()
C. int()
D. format()

Answer: B

Explanation:

class bool([x]) - return a Boolean value, i.e. one of True or False. x is converted using the standard truth testing procedure. If x is false or omitted, this returns False; otherwise, it returns True. The bool class is a subclass of int. It cannot be subclassed further. Its only instances are False and True.
https://docs.python.org/3/library/functions.html#bool

Question: 374

Select the correct ways to create a tuple with one element. (select 3)

A. stocks = 'AAPL'
B. stocks = ('AAPL',)
C. stocks = ('AAPL')
D. stocks = tuple(['AAPL'])
E. stocks = 'AAPL',

Answer: B,D,E

Explanation:

https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences

Question: 375

What is the result of the following code snippet?

```
numbers1 =  [1,2, 3]
numbers2 = numbers1[:]
del numbers1[1:2]
print(numbers1, numbers2)
```

A. [1, 3][1,3]
B. [1, 3][1,2, 3]
C. [1, 2,3][1, 2, 3]
D. [1] [1,2,3]

Answer: B **Explanation:**

[:] - makes a copy.
Steps:
numbers2 = numbers1[:] -> numbers1 = [1, 2, 3], numbers2 = [1, 2, 3] del numbers1[1:2] -> numbers1 = [1, 3], numbers2 = [1, 2, 3]

Question: 376

What does the numbers list look like?
numbers = [[_ for _ in range(i + 1)] for i in range(5)] [[0, 1, 2, 3, 4], [0, 1, 2, 3], [0, 1, 2], [0, 1], [0]]

A. [[0, 1, 2, 3, 4],
[0, 1, 2,3,4],
[0, 1, 2,3,4],
[0, 1, 2,3,4],
[0, 1, 2,3,4]]
B. [[0], [0, 1], [0, 1, 2], [0, 1, 2, 3], [0, 1, 2, 3, 4]]
C. [[], [0], [0, 1], [0, 1, 2], [0, 1, 2, 3]]

Answer: B

Explanation:
[_ for _ inrange(0 + 1)] -> [0]
[_ for _ inrange(1 + 1)] -> [0, 1]
[_ for _ inrange(2 + 1)] -> [0, 1,2]
[_ for _ inrange(3 + 1)] -> [0, 1,2,3]
[_ for _ inrange(4 + 1)] -> [0, 1,2,3, 4]
[[_ for _ in range(i + 1)] for i in range(5)] -> [[0], [0, 1], [0, 1, 2], [0, 1, 2, 3], [0, 1, 2, 3, 4]]

Question: 377

What is the easiest way to access an index when iterating through a list using a for loop?

A. Create a second list with indexes.
B. Create a tuple with indexes.
C. Use the built-in function enumerate().
D. Use the built-in function range().

Answer: C

Explanation:
enumerate(iterable, start=0) - return an enumerate object. iterable must be a sequence, an iterator, or some other object which supports iteration. The next () method of the iterator returned by enumerate() returns a tuple containing a count (from start which defaults to 0) and the values obtained from iterating over iterable.
https://docs.python.org/3/library/functions.html#enumerate

Question: 378

Is Python a dynamically typed language?

A. Yes. In Python, variables have no statically assigned types. There is no need to declare variable types.
B. No

Answer: A

Explanation:
Example: var = 'py' print(var) var = 10 print(var) Output: py 10

Question: 379

Select all true statements. (select 2)

A. Python is a good choice for creating and executing tests for applications.
B. Python is free, open-source, and multiplatform.
C. Python is a good choice for low-lewel programming.
D. Python 3 is backwards compatible with Python 2.

Answer: A,B

Explanation:
Python is a good choice for low-lewel programming. -> False
Python 3 is backwards compatible with Python 2. -> False

Question: 380

What is the difference between is operator and == operator?

A. The is operator is a test for the identity of an object, while == is a value comparison. If we use is, the result will be true if and only if the objects being compared are the same object. If we use == the result will be true every time the values of the compared objects are the same.
B. The == operator is a test for the identity of an object, while is is a value comparison. If we use ==, the result will be true if and only if the objects being compared are the same object. If we use is the result will be true every time the values of the compared objects are the same.
C. There is no difference, these operators can be used interchangeably.

Answer: A

Explanation:
The is operator is a test for the identity of an object, while == is a value comparison.

Question: 381

What is a namespace?

A. A namespace is the entire collection of text variables defined while the program is running.
B. A namespace is a collection of all Python keywords that we shouldn't use when defining variables, functions, classes, etc.
C. A namespace is a naming system used to ensure that names are unique to avoid name conflicts. A namespace is actually a dictionary where the keys are the names of the variables and the values of the dictionary are the values of those variables.

Answer: C

Explanation:
A namespace is a mapping from names to objects. Most namespaces are currently implemented as Python dictionaries, but thatâ€™s normally not noticeable in any way (except for performance), and it may change in the future. Examples of namespaces are: the set of built-in names (containing functions such as abs(), and built-in exception names); the global names in a module; and the local names in a function invocation.
https://docs.python.org/3/tutorial/classes.html#python-scopes-and-namespaces

Question: 382

What is global scope?

A. Global scope only appears when you nest functions within other functions.
B. From the moment we start the program in Python, we are in the global scope. Python turns the main script into a module called main that stores the execution of the main program. The namespace of this module is the global scope of the program.
C. Global scope is created when the function is called. Each time we call the function we create a new global scope. By default, names we assign inside functions only exist in the global scope associated with the function call. When the function exits, the global scope is destroyed.

Answer: B

Question: 383

What is the result of the following code snippet? `var = 50`

```
if var == 50:
     print(f'var = {var}')
elif var % 10 == 0:
     print(f'var % 10 =  {var %10}')
elif var > 10:
     print('var > 10')
else:
     print('end')
```

A. var = 50 var % 10 = 0 var > 10
B. var = 50 var % 10 = 0 var > 10 end
C. var = 50
D. end

Answer: C

Question: 384

What is the output of the following snippet?
True + True + 2 * True - False
The specified operations will not be performed and the TypeError will be raised.

A. 3
B. 4
C. False
D. True

Answer: B

Question: 385

Select which use of print() function is incorrect. (select 2)

A. print('sport', 'style', 'outdoor', sep=3)
B. print('sport', 'style','outdoor', sep='#')
C. print(sep='#', 'sport', 'style', 'outdoor')

D. print('Good', 'luck', 'everyone!')
E. print('Mazda' + ' ' + 'CX' + '-' + '5')

Answer: A,C

Explanation:

print(sep='#', 'sport', 'style', 'outdoor')
SyntaxError: positional argument follows keyword argument.
print('sport', 'style', 'outdoor', sep=3)
TypeError: sep must be None or a string, not int.

Question: 386

Select correct answers about the pass statement. (select 2)

A. The pass statement does not exist in Python.
B. The pass statement represents the empty statement. Nothing happens during execution and is used as a placeholder when the statement is syntactically required.
C. The pass statement is used to skip the rest of the code inside the loop for the current iteration only. The loop does not end but continues with the next iteration.
D. The pass statement is used as a placeholder for future code.

Answer: B,D

Explanation:

The pass statement does nothing. It can be used when a statement is required syntactically but the program requires no action.
https://docs.python.org/3/tutorial/controlflow.html#pass-statements

Question: 387

How many asterisk - * will the following snippet send to the console?

```
numbers =  [[i for i in range(10, 16)] for j in range(3)]

for i in numbers:
     for j in i:
        if j % 5 == 0:
           print('*')
```

A. three
B. fifteen
C. five
D. six

Answer: D

Question: 388

What exception will be raised when you import a module/package that is not available in environment?

A. SyntaxError
B. NameError
C. ModuleNotFoundError
D. NotImplementedError

Answer: C

Explanation:

ModuleNotFoundError is raised by import when a module could not be located. It is also raised when None is found in sys.modules.
https://docs.python.org/3/library/exceptions.html#ModuleNotFoundError

Question: 389

What is the output of the following code snippet if the user enters a 0? `try:`

```
     val = input('Enter a value: ')
     print(int(val) / len(val))
except ZeroDivisionError:
     print('ZeroDivisionError raised...')
except:
     print('Bad input...')
```

A. This code will raise the ValueError exception and output Bad input... to the console.
B. 0.0
C. This code will raise the ZeroDivisionError exception and output ZeroDivisionError raised... to the console.
D. This code will raise the TypeError exception and output Bad input... to the console.

Answer: B

Explanation:
Steps:
val = input('Enter a value: ') -> '0'
print(int(val) / len(val)) -> print(0 / len('0')) -> print(0 / 1) -> 0.0

Question: 390

The following list is given:
stocks = ['AMZN', 'TSLA', 'NVDA', 'AAPL', 'MSFT']
Which of the following code is correct (list comprehension + conditional statement)? (select 3)

A. [if ticker.startswith('A') ticker for ticker in stocks]
B. [ticker if ticker.startswith('A') for ticker in stocks]
C. [ticker for ticker in stocksif len(ticker) == 3]
D. [ticker for ticker in stocksif ticker.startswith('A')]
E. [ticker for ticker in stocksif 'A' in ticker]

Answer: C,D,E

Explanation:
Basic syntax: [expression for value in iterable if condition]
https://docs.python.org/3/tutorial/datastructures.html#list-comprehensions

Question: 391

What is the output of the following code snippet?

```
var1 = var2 = 10
var1 = var1 + 1
var2 = var2 - 2

print(Var1 + Var2)
```

A. 20

B. This code will raise the NameError exception. Var1 and Var2 variables are not defined.
C. 19
D. 21
Answer: B
Explanation:
Python is case-sensitive, so Var1 and Var2 is not defined.

Question: 392

Select the correct answer.

A. Dictionaries don't keep the order of items regardless of the Python version.
B. Dictionaries keep the order of items regardless of the Python version.
C. Dictionaries keep the insertion order since Python 3.7 or higher.
Answer: C

Question: 393

The following variable is defined:
height = -20
What will happen when executing the following code:
assert height > 0

A. The assert statement checks if the value of the height variable is greater than 0. If the condition is false (and in this case it is - it returns False) then an AssertionErorr will be raised with no additional message.
B. The assert statement checks that the value of the height variable is greater than 0. If the condition is false (and in this case it is - it returns False), no error will be raised and code execution continues on the next line (if any).
C. The assert statement checks if the value of the height variable is greater than 0. If the condition is true (and in this case it is - it returns True) then an AssertionErorr will be raised without any additional message.
Answer: A
Explanation:
Assert statements are a convenient way to insert debugging assertions into a program: assert_stmt ::= "assert" expression ["," expression]
https://docs.python.org/3/reference/simple_stmts.html#the-assert-statement

Question: 394

What built-in function can you use to read an attribute/method of a specific object?

A. attr()
B. hasattr()
C. getattr()
D. setattr()
Answer: C
Explanation:
getattr(object, name[, default]) - return the value of the named attribute of object. name must be a string. If the string is the name of one of the objectâ€™s attributes, the result is the value of that attribute. For

example, getattr(x, 'foobar') is equivalent to x.foobar. If the named attribute does not exist, default is returned if provided, otherwise AttributeError is raised.
https://docs.python.org/3/library/functions.html#getattr

Question: 395

What is an object id in Python?

A. An object id is a float number that guarantees that it will be unique and constant for that object throughout its lifetime.
B. An object id is an integer and does not guarantee that it will be unique and constant for that object throughout its lifetime.
C. An object id is an integer that guarantees that it will be unique and constant for that object throughout its lifetime.

Answer: C

Explanation:
id(object) - return the â€œidentityâ€• of an object. This is an integer which is guaranteed to be unique and constant for this object during its lifetime. Two objects with non-overlapping lifetimes may have the same id() value.

https://docs.python.org/3/library/functions.html#id

Question: 396

The following list is given:
artists = ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
What does the list below look like?
result = [len(artist) for artist in artists]

A. [5, 12, 10, 5, 5]
B. ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
C. [True, True, True, True, True]
D. 37
E. []

Answer: A

Explanation:
len(s) - return the length (the number of items) of an object. The argument may be a sequence (such as a string, bytes, tuple, list, or range) or a collection (such as a dictionary, set, or frozen set).
https://docs.python.org/3/library/functions.html#len

Question: 397

Suppose you have the following snippet of code:

```
stocks = [
    "Apple",
    "Tesla",
    "Amazon",
    "Microsoft",
    "Netflix",
    "Alphabet",
```

```
]
tmp = stocks[:]
```

Select all true statements. (select 2)
(Correct)stocks and tmp are different list.

A. stocks and tmp are different names of the same list.
B. stocks is a list and tmp is a tuple.
C. stocks has the same length as tmp.

Answer: C

Explanation:
stocks[:] - return a shallow copy of the list. Equivalent to stocks.copy().

Question: 398

What is the expected behavior of the following program?

```
i = 1
sum = 0

while True and i < 10:
      sum += i
      i += 1

print(sum)
```

A. This code will enter an infinite loop.
B. This code will output 45 to the console.
C. This code will output 0 to the console.
D. This code will output 55 to the console.

Answer: B

Question: 399

What is the output of the following print functions?

```
print('open', end=',')
print('high', end=',')
print('low', end=',')
print('close', end=',')
```

A. open high low close
B. openhighlowclose
C. open,high,low,close,

Answer: C

Explanation:
print(*objects, sep=' ', end='
', file=sys.stdout, flush=False) - print objects to the text stream file, separated by sep and followed by end. sep, end, file, and flush, if present, must be given as keyword arguments.
All non-keyword arguments are converted to strings like str() does and written to the stream, separated by sep and followed by end. Both sep and end must be strings; they can also be None, which means to use the default values. If no objects are given, print() will just write end.

https://docs.python.org/3/library/functions.html#print

Question: 400

What is the result of the following operation?
['A', 'B', 'C'] * 3

A. ['A', 'B', 'C', 'A', 'B','C', 'A','B', 'C']
B. ['A', 'B', 'C', 3]
C. ['A', 'B', 'C']
D. Theoperation cannot beperformed. You cannot multiply an object of type list by an object of type int.
E. ['AAA', 'BBB', 'CCC']

Answer: A

Question: 401

What does PSF do?

A. The PSF organization is a profit-oriented organization dedicated to the development of the Python language.
B. The PSF organization is a non-profit organization dedicated to the development of the Python language. The mission of the foundation is to promote, protect and develop Python, as well as support the international community of programmers writing in this language.
C. The PSF organization implements everything related to the Python language.
D. The PSF organization is only responsible for organizing the Python conferences.

Answer: B

Explanation:
The Python Software Foundation (PSF) is a 501(c)(3) non-profit corporation that holds the intellectual property rights behind the Python programming language.
https://www.python.org/psf/

Question: 402

Is Python case sensitive?

A. Yes
B. No, but you can turn it on.
C. No

Answer: A

Explanation:
Example:
var = 1
Var = 2
print(var == Var) -> False

Question: 403

Which of the following values is of float type? (select 4)

A. 0.0
B. 1 1e6 .0 0.
C. 0
D.
E.
F.

Answer: A,C,D,E **Explanation:**
0 and 1 are of type int.

Question: 404

CPython is...

A .... a compiled language used to perform high-level programming functions.

B .... a programming language that is a superset of Python, designed to produce C-like performance with code written in Python.
C ... another name for Cython, a superset of the Python programming language.
D ... the default implementation of the Python programming language.

Answer: D

Question: 405

The following sets are given:
junior = {'python', 'html', 'css', 'git'}
senior = {'python', 'html', 'css', 'django', 'javascript'}
How do you find the union of these sets? (select 3)

A. junior.union(senior)
B. senior.union(junior)
C. junior + senior
D. junior | senior

Answer: A,B,D

Explanation:
union(*others)
set | other | ...
Return a new set with elements from the set and all others.
https://docs.python.org/3/library/stdtypes.html#frozenset.union

Question: 406

What is the result of the following code snippet? `def compute(i, j): return i ** 2 + j ** 2`

```
print(compute(j=2, 2))
```

A. This code will raise the SyntaxError -> keyword argument follows positional argument.
B. This code will output 8 to the console.
C. This code will raise the SyntaxError -> positional argument follows keyword argument.
D. This code will output None to the console.

Answer: C

Question: 407

What is the result of the following code snippet?

```
try: val = 10 / 0
     print(val) break
except (ValueError, ZeroDivisionError):
     print('ValueError or ZeroDivisionError exception raised...') except:
     print('Something went wrong...')
```

The code will output 0 to the console.

A. This code will raise a ValueError exception and output the following message: 'ValueError or ZeroDivisionError exception raised...'
B. This code will raise a ZeroDivisionError exception and output the following message: 'ValueError or ZeroDivisionError exception raised...'
C. The code will have an exception handled by the last except block.

Answer: B

Question: 408

What is the output of the following print function?

A. print('calculate', 'mean', sep='_')
B. calculate_mean_
C. _calculate_mean_
D. calculatemean_
E. calculate_mean
F. calculate mean

Answer: E

Explanation:
print(*objects, sep=' ', end='
', file=sys.stdout, flush=False) - print objects to the text stream file, separated by sep and followed by end. sep, end, file, and flush, if present, must be given as keyword arguments.
All non-keyword arguments are converted to strings like str() does and written to the stream, separated by sep and followed by end. Both sep and end must be strings; they can also be None, which means to use the default values. If no objects are given, print() will just write end.
https://docs.python.org/3/library/functions.html#print

Question: 409

What is the output of the following code snippet if the user enters 8 and 5 respectively?
var1 = input()
var2 = int(input())
print(var1 * var2)

A. 88888
B. This code will raise the SyntaxError exception.
C. 85
D. This code will raise the TypeError exception.
E. 40
F. 55555555

Answer: A

Explanation:
var1 = input() -> '8'
var2 = int(input()) -> 5
print(var1 * var2) -> print('8' * 5) -> print('88888')

Question: 410

What is the result of the following code snippet?

```
var1 = var2 = 1 var3 = var4 = 0

result1 = var1 ^ var2 result2 = var1 ^ var3 result3 = var1 | var3 result4 = var3 | var4

print(result1, result2, result3, result4)
```

A. 0 1 1 0
B. 1 0 0 1

C. 1 1 10
D. 0 1 11
Answer: A

Question: 411

What is a Python function?

A. A function is a block of code that is only executed when the function is called. To define a function, use the func keyword.
B. A function is a block of code that is only executed when the function is called. To define a function, use the def keyword.
C. A function is a block of code that is only executed when the function is called. To define a function, use the function keyword.
Answer: B
Explanation:
https://docs.python.org/3/tutorial/controlflow.html#defining-functions

Question: 412

The following list is given:

A. net_profit = [-10.5, 4.5, 30.8, -3.5, 14.0]
B. Which of the following code is correct for combining the list comprehension and the conditional statement with the else clause? (select 2)
C. [value if value >= 0 else abs(value) for value in net_profit]
D. [value for value in net_profit else 0]
E. [value for value in net_profit if value >= 0 else 0]
F. [value if value >= 0 else 0 for value in net_profit]
Answer: C,F
Explanation:
List comprehension + if statement basic syntax: [expression if condition else expression for value in

iterable]

Question: 413

What is a local variable?

A. A local variable is a variable that isdeclared using the global keyword.
B. A local variable is a variable that isdeclared using the nonlocal keyword.
C. A local variable is a variable that is available in the local scope.
D. A local variable is a variable that isavailable in the global scope.

Answer: C

Question: 414

What is the output of the following print function? print('*' * 1, '*' * 2, '*' * 4, '*' * 8, sep='#', end='#')

A. #**#**** ******** #
B. #**#**** ********
C. This code will raise the SyntaxError.
D. 1#22#4444#88888888#

Answer: A

Explanation:
print(*objects, sep=' ', end='
', file=sys.stdout, flush=False) - print objects to the text stream file, separated by sep and followed by end. sep, end, file, and flush, if present, must be given as keyword arguments.
All non-keyword arguments are converted to strings like str() does and written to the stream, separated by sep and followed by end. Both sep and end must be strings; they can also be None, which means to use the default values. If no objects are given, print() will just write end.
https://docs.python.org/3/library/functions.html#print

Question: 415

The following sets are given:
junior = {'python', 'html', 'css', 'git'}
senior = {'python', 'html', 'css', 'django', 'javascript'}
How do you find the intersection of these sets? (select 3)

A. junior * senior
B. junior & senior
C. junior.intersection(senior)
D. senior.intersection(junior)

Answer: B,C,D

Explanation:
intersection(*others) or set & other & ... - return a new set with elements common to the set and all others.
https://docs.python.org/3/library/stdtypes.html#frozenset.intersection

Question: 416

The following list of tuples is given:

data = [(1,2,3), (4,5,6), (7,8,9)]
How do you sort this list by the second element of each tuple? (select 2)

A. New sorted list: sorted(data, key=lambda tup: tup[1])
B. Inplace: data.sort(key=lambda tup: tup)
C. Inplace: data.sort(key=lambda tup: tup[1])
D. New sorted list: sorted(data, key=lambda tup: tup[0])

Answer: A,C

Explanation:
https://docs.python.org/3/howto/sorting.html#sorting-basics

Question: 417

What built-in function will you use to find the object id?

A. bool()
B. type()
C. id()
D. object()

Answer: C

Explanation:
id(object) - return the â€œidentityâ€• of an object. This is an integer which is guaranteed to be unique and constant for this object during its lifetime. Two objects with non-overlapping lifetimes may have the same id() value.
https://docs.python.org/3/library/functions.html#id

Question: 418

What is the expected behavior of the following program?

```
i =     0
flag = False

while True: i +=    1
     if i <  0:
          flag = True
          break

print(flag)
```

A. This code will not make a single pass through the loop.
B. This code will output True to the console.
C. This code will enter an infinite loop.
D. This code will output False to the console.

Answer: C

Explanation:
break_stmt ::= "break"
break may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop.
https://docs.python.org/3/reference/simple_stmts.html#the-break-statement

Question: 419

Suppose you have the following function: def compute(i, j, k):
return i ** 2 + j ** 2 + k ** 2
Which function call is correct? (select 3)

A. compute(k=4, j=3, i=2)
B. compute(i=2, j=3, k=4)
C. compute(i=2, 3, k=4)
D. compute(2, 3, 4)
E. compute(i=2, j=3, 4)

Answer: A,B,D

Explanation:
compute(i=2, j=3, 4)
SyntaxError: positional argument follows keyword argument compute(i=2, 3, k=4)
SyntaxError: positional argument follows keyword argument

Question: 420

Select all true statements about dictionaries in Python (select 2)

A. Tuple can be a key in a dictionary.
B. Set can be a key in a dictionary.
C. List can be a key in a dictionary.
D. It's not possible to have more than one key of the same value in a dictionary.

Answer: A,D

Explanation:
List can be a key in a dictionary. -> False, list is a mutable type.
Set can be a key in a dictionary. -> False, set cannot be a key in a dictionary.

Question: 421

What exception will be raised if we try to divide by 0?

A. OSError
B. ZeroDivisionError
C. MemoryError
D. RuntimeError

Answer: B

Explanation:
ZeroDivisionError is raised when the second argument of a division or modulo operation is zero. The associated value is a string indicating the type of the operands and the operation.
https://docs.python.org/3/library/exceptions.html#ZeroDivisionError

Question: 422

What is the output of the following snippet? var1 = var2 = var3 = 10 print(var1, var2, var3, sep='-')

A. 10 10 10
B. True-True-10

C. var1-var2-var3
D. 10-10-10
Answer: D

Question: 423

A list of numbers is given:
numbers = list(range(10))
The following operation was performed:
numbers[:5] = [sum(numbers[:5])]
What does the numbers list look like?
Unable to make this assignment, a TypeError will be raised.

A. [10]
B. [15, 5, 6, 7, 8, 9]
C. [10, 5, 6, 7, 8, 9]
Answer: C
Explanation:
Steps:
numbers = list(range(10)) -> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[sum(numbers[:5])] -> [sum([0, 1, 2, 3, 4])] -> [10]
numbers[:5] = [sum(numbers[:5])]
numbers -> [10, 5, 6, 7, 8, 9]

Question: 424

What is the result of the following code snippet?

```
def func1(x, y):
     result = x + y + 2 * x * y
     return None

def func2():
     return func1(1, 2) * func1(2, 1)

print(func2())
```

A. Thiscode will output 49 to the console.
B. Thiscode will output 7 to the console.
C. Thiscode will cause a SyntaxError exception.
D. Thiscode will cause a TypeError exception.
Answer: D

Question: 425

A newline character in Python is:

A.
B.
C.
D.
E. ew
F.

Answer: A

Question: 426

The following variable is given: numbers = [i for i in range(-10, -15)] Select all true statements. (select 2)

A. type(numbers) == tuple
B. len(numbers) == 0
C. len(numbers) == 5
D. type(numbers) == list

Answer: B,D

Explanation:

The range type represents an immutable sequence of numbers and is commonly used for looping a specific number of times in for loops.
https://docs.python.org/3/library/stdtypes.html#range

Question: 427

What is a Python module?

A. A module is a directory that contains Python scripts.
B. A module is a file containing Python code (classes, functions variables, etc.). The module name is the name of the file without the .py extension. In the module, the module name (as a string) is available as the global variable name .
C. A module is a file containing Python code (classes, functions variables, etc.). In the module, the name of the module (as a string) is available as the global variable name.
D. A module is another name for a function.

Answer: B

Explanation:

A module is a file containing Python definitions and statements. The file name is the module name with the suffix .py appended. Within a module, the moduleâ€™s name (as a string) is available as the value of the global variable name .
https://docs.python.org/3/tutorial/modules.html

Question: 428

The following dictionary is given:
eng_to_spa = {'one': 'uno', 'two': 'dos', 'three': 'tres'}
How do you replace the values with the keys of this dictionary?

A. {val,key forkey,val in eng_to_spa.items()}
B. {val:key forkey,val in eng_to_spa}
C. {val: key for key, val in eng_to_spa.items()}
D. {{val: key} for key, val in eng_to_spa.items()}

Answer: C

Question: 429

You want to print to the console the following message: 'Hello! I'm happy to be here.'

How can you do that? (select 3)

A. print("""Hello! I'm happy to be here.""")
B. print('Hello! I\'m happy to be here.')
C. print('Hello! I/'m happy to be here.')
D. print('Hello! I'm happy to be here.')
E. print("Hello! I'm happy to be here.")

Answer: A,B,E

Question: 430

What is scope?

A. A scope is a built-in function in Python to return the length of an object.
B. A scope is a predetermined range for integers.
C. Every Python object is within a certain scope. A scope is the piece of code in which an object in Python remains available.
D. A scope is a built-in data type in Python.

Answer: C

Explanation:

A scope is a textual region of a Python program where a namespace is directly accessible. â€œDirectly accessibleâ€• here means that an unqualified reference to a name attempts to find the name in the namespace.
https://docs.python.org/3/tutorial/classes.html#python-scopes-and-namespaces

Question: 431

What is the difference between / and // operators?

A. The / operator does a normal division, and the // operator does an integer division.
B. The // operator does a normal division, and the / operator does an integer division.
C. There are no such operators in Python.
D. These operators can be used interchangeably.

Answer: A

Explanation:

The / (division) and // (floor division) operators yield the quotient of their arguments. The numeric arguments are first converted to a common type. Division of integers yields a float, while floor division of integers results in an integer; the result is that of mathematical division with the â€˜floorâ€™ function applied to the result.

Question: 432

Consider the basic form of the following instruction: `try: ... except:`

Select correct statement. (select 1)

A. Python tries to execute the code that follows the try statement as normal part of the program. When an exception is raised in the try block, the code following the except statement is executed.

B. Python tries to execute the code that follows the try statement as normal part of the program. If an exception is not raised in the try block, the code following the except statement is executed.
C. Python tries to execute the code that follows the try statement as normal part of the program. When an exception is raised in a try block, the code following the except statement is not executed.

Answer: A

Explanation:

The try statement specifies exception handlers and/or cleanup code for a group of statements:
try_stmt ::= try1_stmt | try2_stmt
try1_stmt ::= "try" ":" suite
("except" [expression ["as" identifier]] ":" suite)+
["else" ":" suite]
["finally" ":" suite]
try2_stmt ::= "try" ":" suite
"finally" ":" suite
https://docs.python.org/3/reference/compound_stmts.html#the-try-statement

Question: 433

The following Python code is given as a string:
code = 'import math; print(math.pi)'
What built-in function can you use to execute this code?

A. exec()
B. repr()
C. enumerate()
D. eval()
E. compile()

Answer: A

Explanation:

exec(object[, globals[, locals]]) - this function supports dynamic execution of Python code. object must be either a string or a code object.
https://docs.python.org/3/library/functions.html#exec

Question: 434

What is the output of the following code snippet if the user enters a 0?

```
try: val = int(input('Enter a value: ')) print(val / len(val))
except ZeroDivisionError:
    print('ZeroDivisionError raised...') except:
    print('Bad input...')
```

A. This code will raise the TypeError exception and output Bad input... to the console.
B. This code will raise the ZeroDivisionError exception and output ZeroDivisionError raised... to the console.
C. 0.0
D. This code will raise the ValueError exception and output Bad input... to the console.

Answer: A

Explanation:

val = int(input('Enter a value: ')) -> '0'

print(val / len(val)) -> print('0' / len('0')) -> It will raise the TypeError exception.

Question: 435

What is the output of the following code snippet?

```
numbers =  (10, 20, 30, 40, 50)
numbers = numbers[-3:-2]
numbers = numbers[-1]
print(numbers)
```

A. 30
C. (30,)
C. 20
D. This code will raise the IndexError exception.
Answer: A
Explanation:
Steps:
numbers = numbers[-3:-2]
numbers -> (30,)
numbers = numbers[-1]
numbers -> 30

Question: 436

The following dictionary is given:
stocks = {'MSFT': 280.0, 'AAPL': 146.39, 'TSLA': 644.22} How do you get all the keys of this dictionary?

A. stocks.keys()
B. stocks.get_keys()
C. stocks.values()
D. stocks.keys
E. stocks.items()
Answer: A
Explanation:
dict.keys() - return a new view of the dictionarya€™s keys.
https://docs.python.org/3/library/stdtypes.html#dict.keys

Question: 437

The following operations were performed:
numbers = [1, 2, 3]
numbers.append(numbers)
Select the form of the numbers list.

A. [1, 2, 3]
B. The same list cannot be appended to itself.
C. []
D. [1, 2, 3, [...]]
Answer: D
Explanation:
list.append(x) - add an item to the end of the list. Equivalent to a[len(a):] = [x].

https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 438

The following variable is defined:
number = 10
Which syntax for a conditional statement with the else clause in one line is correct? (select 2)

A. 'positive' else 'negative' if number >= 0
B. print('positive') if number >= 0 else print('negative')
C. if number >= 0: 'positive' else 'negative'
D. 'positive' if number >= 0 else 'negative'

Answer: B,D

Explanation:
Basic syntax: [expression if condition else expression]

Question: 439

What does **kwargs mean in the function?

A. Python does not have a **kwargs syntax.
B. The special syntax **kwargs is most often used in Python function definitions, which is used to pass a variable number of arguments to a function (arguments that do not have a specific name).
C. The special syntax **kwargs is most often used in Python function definitions, which is used to pass a variable number of arguments to a function (arguments with specific names).

Answer: C

Explanation:
**kwargs specifies that arbitrarily many keyword arguments can be provided to a function.

Question: 440

We need to check if all elements of the iterable object evaluate to the truth. What built-in function can you use for this?

A. any()
B. all()
C. bin()
D. bool()

Answer: B

Explanation:
all(iterable) - return True if all elements of the iterable are true (or if the iterable is empty).
https://docs.python.org/3/library/functions.html#all

Question: 441

The Style Guide for Python Code recommends a naming convention for variables and functions. Select all true statements. (select 2)

A. Function names follow the same convention as variable names.
B. Variable names should be lowercase, with words separated by underscores to improve readability

(e.g., rate, learning_rate).
C. You should use camelCase for class names (e.g., regression, linearRegression).
D. Function names follow the same convention as class names.
Answer: A,B
Explanation:
You should use PascalCase for class names (e.g., Regression, LinearRegression).

Question: 442

How many hashes - # will the following snippet send to the console?

```
for i in range(1, 10): if i ==    4:
        break

    print('#')
```

A. three
B. nine
C. two
D. four
Answer: A
Explanation:
break_stmt ::= "break"
break may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop.
https://docs.python.org/3/reference/simple_stmts.html#the-break-statement

Question: 443

What is the difference between list.remove() and list.pop()?

A. There is no difference.
B. The list.remove() method removes the first occurrence of the matching object. The list.pop() method removes and returns an object by index. If no index is passed to list.pop(), the last element of the list is removed and returned.
C. The list.pop() method removes the first occurrence of the matching object. The list.remove() method removes and returns an object by index. If no index is passed to list.remove(), the last element of the list is removed and returned.
Answer: B
Explanation:
list.remove(x)
Remove the first item from the list whose value is equal to x. It raises a ValueError if there is no such item.
list.pop([i])
Remove the item at the given position in the list, and return it. If no index is specified, a.pop() removes and returns the last item in the list.

Question: 444

How can you assign number 0.00036 to the variable in Python? (select 2)

A. var =36e-4
B. var =3.6e4
C. var =0.00036
D. var =3.6e-4

Answer: C,D

Explanation:
var = 3.6e4 -> 36000.0
var = 36e-4 -> 0.0036

Question: 445

The following variable is defined:
height = -20
What will happen when executing the following code:
assert height > 0, 'height must be positive'

A. The assert statement checks if the value of the height variable is greater than 0. If the condition is false (and in this case it is - it returns False) then an AssertionError will be raised without any additional message.
B. The assert statement checks that the value of the height variable is greater than 0. If the condition is false (and in this case it is - it returns False), no error will be raised and code execution continues on the next line (if any).
C. The assert statement checks if the value of the height variable is greater than 0. If the condition is false (and in this case it is - returns False) the AssertionError will be raised with the appropriate message 'height must be positive'.

Answer: C

Explanation:
Assert statements are a convenient way to insert debugging assertions into a program: assert_stmt ::= "assert" expression ["," expression]
https://docs.python.org/3/reference/simple_stmts.html#the-assert-statement

Question: 446

What operator is used to check the equality of objects?

A. =
B. ===
C. ==
D. !=
E. <>

Answer: C

Explanation:
https://docs.python.org/3/reference/expressions.html#value-comparisons

Question: 447

Which variables below are incorrectly defined? (select 4)

A. year = [24, 53, 65, 12]
B. _year = [24, 53, 65, 12]
C. class = 'English Course'
D. var1 = 100
E. def = 'This is a short definition of sth.'
F. if = 30
G. 2020_year = [24, 53, 65, 12]

Answer: C,E,F,G

Explanation:

The variable name cannot start with a digit.
if, class, def - these are keywords in Python and cannot be used as variable names.

Question: 448

Which of the following is an example of a Python file extension?

A. py
B. c
C. p
D. pyt

Answer: A

Question: 449

The meaning of the keyword parameter is determined by...

A. its connection with existing variables.
B. its value.
C. its position within the argumentlist.
D. the argument's name specified along with its value.

Answer: D

Question: 450

What is the built-in function len() for?

A. The len() function returns a list of the attributes and methods of a given object.
B. The len() function returns information about the memory used by a given object.
C. Python does not have a built-in function called len().
D. The len() function returns the number of elements in a collection/sequence.

Answer: D

Explanation:

len(s) - return the length (the number of items) of an object. The argument may be a sequence (such as a string, bytes, tuple, list, or range) or a collection (such as a dictionary, set, or frozen set).
https://docs.python.org/3/library/functions.html#len

Question: 451

How many hashes - # will the following snippet send to the console? `i = 0`

```
while True and i <= 7:
    if i % 3 == 0:
```

```
            i += 1
            continue
        print(i, '#')
        i += 1
```

A. four

B. five zero six
C. **Answer: B**
D. **Explanation:**
continue_stmt ::= "continue"
continue may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop. It continues with the next cycle of the nearest enclosing loop.
https://docs.python.org/3/reference/simple_stmts.html#the-continue-statement

Question: 452

Which operator is used for exponentiation in Python?

A. *
B. ^
C. >>
D. **
E. //

Answer: D
Explanation:
** - exponentiation

Question: 453

Can list comprehension be nested?

A. No
B. Yes

Answer: B
Explanation:
Example:
[i for i in range(4) for j in range(3)]
https://docs.python.org/3/tutorial/datastructures.html#nested-list-comprehensions

Question: 454

What is the output of the following code snippet?

```
def introduce(class): return f'Hello from {class} class.'

print(introduce('Bachata'))
```

A. This code will raise the SyntaxError exception. You cannot use the keyword class as the name of a function argument.
B. This code will raise the SyntaxError exception. You cannot use string formatting like this.
C. This code will output Hello from Bachata class. to the console.
D. This code will output Hello from {class} class. to the console.

Answer: A

Question: 455

The following dictionary is given:
Sstocks = {'MSFT': 280.0, 'AAPL': 146.39, 'TSLA': 644.22} How to get all the values of this dictionary?

A. stocks.values
B. stocks.get_values()
C. stocks.values()
D. stocks.value()
E. stocks.keys()

Answer: C

Explanation:
dict.values() - return a new view of the dictionarya€™s values.
https://docs.python.org/3/library/stdtypes.html#dict.values

Question: 456

This question is a tricky one. What is the result of the following code snippet?

```
numbers = [1,    2,
          3,  4, 5]
result =  []

for number in numbers:
     result.append(numbers.pop())

print(result)
```

A. [1,2,3,4, 5]
B. [3,4,5]
C. [5,4,3]
D. [5,4,3,2, 1]

Answer: C

Explanation:
It is not best practice to iterate over the object from which we are removing items in this way in Python. We should use [:] to copy this list to avoid any confusion. For exaample: numbers = [1, 2, 3, 4, 5] result = [] for number in numbers[:]: result.insert(0, numbers.pop()) print(result)

Question: 457

What are keywords in Python?

A. Keywords are Python reserved words. We cannot use the keyword as a variable name, function name, or any other identifier. They are used to define Python syntax and structure.
B. Keywords are all the keys of the dictionary.
C. Keywords are used to define Python syntax and structure, and we can use them as variable names or function names.

Answer: A

Explanation:
https://docs.python.org/3/reference/lexical_analysis.html#keywords

Question: 458

What is the expected behavior of the following program?

```
i = 0
flag = False

while True: if i >= 10:
          flag = True break i += 1

print(i)
```

A. This code will output 10 to the console.
B. This code will enter an infinite loop.
C. This code will output 11 to the console.
D. This code will output True to the console.

Answer: A

Explanation:

break_stmt ::= "break"
break may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop.
It terminates the nearest enclosing loop, skipping the optional else clause if the loop has one.
https://docs.python.org/3/reference/simple_stmts.html#the-break-statement

Question: 459

What keyword is used to define functions in Python?

A. function
B. def
C. func
D. void
E. class

Answer: B

Explanation:

The keyword def introduces a function definition. It must be followed by the function name and the parenthesized list of formal parameters. The statements that form the body of the function start at the next line, and must be indented.

https://docs.python.org/3/tutorial/controlflow.html#defining-functions

Question: 460

What is a decorator?

A. A decorator is an object that we can iterate over.
B. A decorator is another form of comments that is taken into account when creating documentation.
C. A decorator is a built-in data type in Python.
D. A decorator is a design pattern in Python that allows the user to add

new functionality to anexisting object without modifying its structure.

Answer: D

Question: 461

What is the output of the following code snippet if the user enters 3 and 7 respectively? num1 = input() num2 = input() print(num1 + num2)

A. 10
B. 37
C. None
D. This code will raise the TypeError exception.

Answer: B

Explanation:

Steps:
num1 = input() -> '3'
num2 = input() -> '7'
print(num1 + num2) -> print('3' + '7') -> print('37')

Question: 462

What is the meaning of the following code snippet that is often placed at the end of a Python script? if name == ' main ':

A. This is a conditional statement that allows you to control the running of a certain piece of code depending on how the script is run (directly, indirectly). The built-in variable name stores the name of the current module. If the module is run directly, it takes the value main . If a module is run indirectly, the built-in variable name takes the name of the module.
B. This is a conditional statement that allows you to control the running of a certain piece of code depending on how the script is run (directly, indirectly). The built-in variable name stores the name of the current module. If the module is run indirectly, it takes the value main . If a module is run directly, the built-in variable name takes the name of the module.

Answer: A

Question: 463

The following list is given:
numbers = ['1', '2', '43', '6', '2']
How do you convert the list items to int type? (select 1)

A. [int(number) for number in numbers] B. [str(number) for number in numbers] C. [float(number) for number in numbers] D. numbers.convert_to(int)

Answer: A

Question: 464

What is the expected output of the following code? course_name = 'Python in Data Science' tokens =

course_name.split() print(tokens[-2:])

A. ['in', 'Data', 'Science']
B. ['Python', 'in']
C. ['c', 'e']
D. ['Data', 'Science']

Answer: D

Explanation:
tokens -> ['Python', 'in', 'Data', 'Science']

Question: 465

How can you create a single-line comment in Python?

A. Pythonsingle-linecomments beginwith the /* characters
B. Pythonsingle-linecomments beginwith a/ character
C. Pythonsingle-linecomments beginwith a* character
D. Pythonsingle-linecomments beginwith a# character
E. You cannot create single-line comments in Python

Answer: D

Explanation:
Example:
# This is a single-line comment
var = 1
print(var)

Question: 466

How many hashes - # will the following snippet send to the console? for i in range(1, 6):
if i == 4: continue print('#')

A. four
B. zero
C. three
D. five

Answer: A

Explanation:
continue_stmt ::= "continue"
continue may only occur syntactically nested in a for or while loop, but not nested in a function or class definition within that loop. It continues with the next cycle of the nearest enclosing loop.
https://docs.python.org/3/reference/simple_stmts.html#the-continue-statement

Question: 467

What exception will be raised when you try to access an object attribute that is undefined?

A. NameError
C. SyntaxError
D. NotImplementedError
D. AttributeError

E. ReferenceError

Answer: D

Explanation:

AttributeError is raised when an attribute reference (see Attribute references) or assignment fails. (When an object does not support attribute references or attribute assignments at all, TypeError is raised.)
https://docs.python.org/3/library/exceptions.html#AttributeError

Question: 468

The following list is given:
artists = ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
The following operation was performed on the list:
artists.pop()
Select artists list after this operation.

A. []
B. ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
C. ['Phil Collins', 'The Police', 'Queen', 'AC/DC']
D. ['Sting', 'Phil Collins', 'The Police', 'Queen']

Answer: D

Explanation:

list.pop([i]) - remove the item at the given position in the list, and return it. If no index is specified, a.pop() removes and returns the last item in the list.
https://docs.python.org/3/tutorial/datastructures.html#more-on-lists

Question: 469

What built-in function will you use to open a file?

A. connect()
B. open()
C. read()
D. write()

Answer: B

Explanation:

open(file, mode='r', buffering=- 1, encoding=None, errors=None, newline=None, closefd=True, opener=None) - open file and return a corresponding file object. If the file cannot be opened, an OSError is raised.
https://docs.python.org/3/library/functions.html#open

Question: 470

Can we create a dictionary that has two identical keys?

A. No, the keys in dictionary are unique.
B. Yes.

Answer: A

Explanation:

Example:

```
stocks = {'Apple': 150, 'Apple': 700}
print(stocks)
```

Output:

```
{'Apple': 700}
```

Question: 471

We need to check if at least one element of the iterable object evaluates to the truth. What built-in function can you use for this?

A. any()
B. type()
C. bool()
D. all()

Answer: A

Explanation:

any(iterable) - return True if any element of the iterable is true. If the iterable is empty, return False.
https://docs.python.org/3/library/functions.html#any

Question: 472

The following two lists are given: tech_names = ['python', 'django', 'html']
exps = ['senior', 'mid', 'junior']
We want to iterate through these lists simultaneously. What built-in function will you use for this purpose?

A. zip()
B. filter()
C. reduce()
D. map()
E. zip_longest()

Answer: A

Explanation:

zip(*iterables, strict=False) - iterate over several iterables in parallel, producing tuples with an item from each one.
https://docs.python.org/3/library/functions.html#zip

Question: 473

Can we use indexes to read dictionary items (as with lists/tuples)?

A. No, because set is an unordered collection.
B. Yes.

Answer: A

Explanation:

Example:

```
stocks = {'Apple': 150, 'Tesla': 700}
print(stocks[1])
```

It will raise the KeyError exception.

Question: 474

What is PEP?

A. Packet Exchange Protocol
B. PEP stands for Python Enhancement Proposal. The PEP is a design document that provides information to the Python community or describes a new Python function, its processes, or its environment. It is also a set of rules that determine how Python code is formatted for maximum readability.
C. Python Enterprise Project

Answer: B

Question: 475

The following dictionaries are given: stocks_1 = {'CDR': 200.0, 'PLW': 420.0} stocks_2 = {'11B': 510.0, 'CDR': 205.5} What will be the result of the following operation?

A. {**stocks_1, **stocks_2}

B. {'CDR': 205.5}
C. {'11B': 510.0, 'CDR': 200.0, 'PLW': 420.0}
D. {'11B': 510.0, 'CDR': 205.5, 'PLW': 420.0}
E. {'11B': 510.0, 'CDR': 200.0, 'CDR': 205.5, 'PLW': 420.0}

Answer: D

Question: 476

Select true statements. (select 2)

A. An iterator is an object that implements the next () method that returns the next element of the iterable object, or throws a StopIteration exception when there are no more elements available.
B. An iterator is an object that has a iter () method that returns an iterator object.
C. Any function in Python is an iterator.

Answer: A,B

Explanation:
iterator - an object representing a stream of data. Repeated calls to the iteratora€™s next () method (or passing it to the built-in function next()) return successive items in the stream. When no more data are available a StopIteration exception is raised instead. At this point, the iterator object is exhausted and any further calls to its next () method just raise StopIteration again. Iterators are required to have an iter () method that returns the iterator object itself so every iterator is also iterable and may be used in most places where other iterables are accepted.

Question: 477

You want to print to the console the following message: I really like "Monty Python's Flying Circus" series.
How can you do that? (select 3)

A. print("I really like "Monty Python\'s FlyingCircus"series.")
B. print('"I really like "Monty Python's FlyingCircus"series."')
C. print('I really like "Monty Python\'s FlyingCircus"series.')
D. print("I really like "Monty Python\'s FlyingCircus"series.")
E. print('I really like "Monty Python's Flying Circus" series.') **Answer: B,C,D**

Question: 478

The following dictionary is given:
stocks = {'PLW': 400, 'CDR': 200, 'BBT': 25}
How do you get the key for the highest value in a dictionary? (select 3)

A. max(stocks, key=stocks.values)
B. max(stocks, key=lambda ticker: stocks.get(ticker))
C. max(stocks, key=lambda ticker: stocks[ticker])
D. max(stocks, key=stocks.get)

Answer: B,C,D
Explanation:

max(iterable, *[, key, default])
max(arg1, arg2, *args[, key])
Return the largest item in an iterable or the largest of two or more arguments.
https://docs.python.org/3/library/functions.html#max

Question: 479

You have the following function:
def factorial(n):
if n < 0:
return None
if n < 2:
return 1
return n * factorial(n - 1)
What is the result of the following function call?

A. print(factorial(5))
B. 24
C. None
D. 120
E. 720

Answer: D

Explanation:
Steps:
factorial(5) -> 5 * factorial(4) -> 5 * 4 * factorial(3) -> 5 * 4 * 3 * factorial(2) -> 5 * 4 * 3 * 2 * factorial(1) -> 5 * 4 * 3 * 2 * 1 * factorial(0) -> 5 * 4 * 3 * 2 * 1 * 1 -> 120

Question: 480

Suppose you have the following function: def compute(i, j, k=2):
return i ** 2 + j ** 2 + k ** 2
Which function call is correct? (select 2)

A. compute(k=3, i=1, j=4) compute()
B. compute(i=2, j=2)
C. compute(j=3, i=1, 4)
D.

Answer: A,C

Explanation:
compute(j=3, i=1, 4)
SyntaxError: positional argument follows keyword argument
compute()
TypeError: compute() missing 2 required positional arguments: 'i' and 'j'

Question: 481

The part of your code where you think an exception may occur should be placed inside:
A. the finally: branch
B. the else: branch
C. the try: branch
D. the except: branch

Answer: C

Explanation:
The try statement specifies exception handlers and/or cleanup code for a group of statements: try_stmt ::= try1_stmt | try2_stmt
try1_stmt ::= "try" ":" suite
("except" [expression ["as" identifier]] ":" suite)+
["else" ":" suite]
["finally" ":" suite]
try2_stmt ::= "try" ":" suite
"finally" ":" suite
https://docs.python.org/3/reference/compound_stmts.html#the-try-statement

Question: 482

What happens when you try to run the following code? print(Welcome to the world of Python!)

A. This code will raise the SyntaxError exception.
B. This code will print Welcome to the world of Python! to the console.
C. This code will raise the TypeError exception.
D. This code will raise the NameError exception.

Answer: A

Explanation:
Missing quotes.

Question: 483

Can you create your own exception in Python?

A. No
B. Yes

Answer: B

Explanation:
Example:
class MyCustomException(Exception):
pass
Exceptions should typically be derived from the Exception class, either directly or indirectly.
https://docs.python.org/3/tutorial/errors.html#user-defined-exceptions

Question: 484

How to create a global variable from local scope (function)?

A. Use the local keyword.
B. Use the nonlocal keyword.
C. You can't do this in Python.
D. Use the global keyword.

Answer: D

Explanation:
The global statement is a declaration which holds for the entire current code block. It means that the listed identifiers are to be interpreted as globals. It would be impossible to assign to a global variable without global, although free variables may refer to globals without being declared global.

https://docs.python.org/3/reference/simple_stmts.html#the-global-statement

Question: 485

Consider the following try... except... else... instruction: try:

except:
... else:
...

Select correct statement.

A. If the code in the try block executes correctly and no exception is raised, the code after the else statement is not executed.
B. If the code in the try block executes correctly and no exception is raised, the code is executed after the else statement, and only in this case.
C. This instruction is incorrect. except clause should be the last one.

Answer: B

Explanation:
The try statement specifies exception handlers and/or cleanup code for a group of statements:
try_stmt ::= try1_stmt | try2_stmt
try1_stmt ::= "try" ":" suite
("except" [expression ["as" identifier]] ":" suite)+
["else" ":" suite]
["finally" ":" suite]
try2_stmt ::= "try" ":" suite
"finally" ":" suite
https://docs.python.org/3/reference/compound_stmts.html#the-try-statement

Question: 486

What happens when you try to run the following code?

A. print(welcome)
B. This code willprint welcome to the console.
C. This code will raise theSyntaxError exception.
D. This code will raise theNameError exception.
E. This code will raise theTypeError exception.

Answer: D

Question: 487

Does converting a list/tuple to a set keep the order of the elements?

A. No, sets are not ordered collections.
B. Yes

Answer: A

Explanation:
Example:
stocks = ['Apple', 'Tesla', 'Facebook', 'Nvidia']
stocks = set(stocks)

print(stocks)
Output:
{'Facebook', 'Nvidia', 'Tesla', 'Apple'}

Question: 488

What is the output of the following code snippet?

```
def calculate(a): number = 10
    return a * number


number =  100
print(calculate(5))
```

A. 50
B. 5
C. This code will raise the SyntaxError exception.
D. 500

Answer: A

Question: 489

You have the following function:

```
def fibonacci(n):
    if n <  1:
        return None
    if n <  3:
        return 1
    return fibonacci(n -  1) + fibonacci(n -  2)
```

What is the result of the following function call?

A. print(fibonacci(4))
B. 5
C. 2
D. 3
E. 8

Answer: D

Explanation:

Steps:

fibonacci(4) -> fibonacci(3) + fibonacci(2) -> (fibonacci(2) + fibonacci(2)) + fibonacci(2) -> 1 + 1 + 1 -> 3

Question: 490

Can a for loop contain an else clause?

A. Yes
B. No

Answer: A

Explanation:

Example:

```
numbers = [3, 5, 7, -2]
for number in numbers:
if number == 0:
print('Found.')
else:
print('Not found.')
```

https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-lo ops

Question: 491

What the following code returns: all((' ', [[]], True, ((), ())))

A. NoneType None False True
B. **Answer: D**
C. **Explanation:**
D. all(iterable) - return True if all elements of the iterable are true (or if the iterable is empty).
https://docs.python.org/3/library/functions.html#all

Question: 492

Which of the following characters is a Python escape character?

A. /
B. \
C. @
D. n
E. $

Answer: B

Explanation:

The backslash \ is the escape character in Python.

Question: 493

The following list is given:
nested = [['TSLA', 'MSFT', 'AMZN'], ['AAPL', 'NVDA']]
We want to use list comprehension to get a flattened version of the nested list: ['TSLA', 'MSFT', 'AMZN', 'AAPL', 'NVDA']
Select the correct answer.

A. [company for company in companies for companies in nested]
B. [company for companies in nested for company in companies]
C. [[company for company in companies] for companies in nested]
D. [companies for companies in nested]

Answer: B

Explanation:
https://docs.python.org/3/tutorial/datastructures.html#list-comprehensions

Question: 494

Select the correct statements regarding the differences between a normal function and a generator function. (select 2)

A. The generator function contains one or more yield statements.
B. When invoked, the generator function returns an iterable object automatically implementing methods such as iter () and next ().
C. There is no difference between a normal function and a generator function.

Answer: A,B

Question: 495

What happens when a dictionary is converted to a set using set()?

A. The result is a set of dictionary keys.
B. The result is a set of dictionary values.
C. You cannot convert a dictionary to a set.
D. The result is a set of tuples (key, value) of a given dictionary.

Answer: A

Explanation:
class set([iterable]) - return a new set object, optionally with elements taken from iterable.
https://docs.python.org/3/library/functions.html#func-set

Question: 496

What types of loops are available in Python? (select 2)

A. for
B. while
C. do until
D. do while

Answer: A,B

Question: 497

What is the result of the following code snippet? techs = tuple('pandas') + ('sql',) print(techs)

A. This code will output ('p', 'a', 'n', 'd', 'a', 's', 's', 'q', 'l') to the console.
B. This code will output ('p', 'a', 'n', 'd', 'a', 's', 'sql') to the console.
C. This code will raise the TypeError exception.
D. This code will output ('pandas', 'sql') to the console.

Answer: B

Explanation:
Steps:
tuple('pandas') -> ('p', 'a', 'n', 'd', 'a', 's')
tuple('pandas') + ('sql',) -> ('p', 'a', 'n', 'd', 'a', 's', 'sql')

Question: 498

The following list is given:
numbers = [1, 4, 5, 7]
What value will be returned when we want to read the item with index 4?

A. numbers[4]
B. None
C. The KeyError will be raised.
D. The NameError will be raised.
E. 7
F. The IndexError will be raised.

Answer: F

Explanation:
IndexError is raised when a sequence subscript is out of range. (Slice indices are silently truncated to fall in the allowed range; if an index is not an integer, TypeError is raised.)
https://docs.python.org/3/library/exceptions.html#IndexError

Question: 499

What statement is used to raise the specified error/exception?

A. raise
B. finally
C. try
D. assert
E. async

Answer: A

Explanation:
https://docs.python.org/3/tutorial/errors.html#handling-exceptions

Question: 500

Which of the following operators is not an assignment operator?

A. /=
B. +=

C. =
D. *=
E. ==
F. -=

Answer: E

Explanation:
== - equality comparison operator

Question: 501

What is the result of the following code snippet?

```
ds_techs =  ('sql', ) + ('pandas', )
dev_techs =  ('git', ) + ('aws', )
techs = ds_techs + dev_techs print(techs)
```

A. This code will output None to the console.
B. This code will raise the TypeError exception.
C. This code will output ('sql', 'pandas', 'git', 'aws') to the console.
D. This code will output ('sql',), ('pandas',), ('git',), ('aws',) to the console.

Answer: C

Explanation:
Steps:
ds_techs = ('sql',) + ('pandas',) -> ('sql', 'pandas')
dev_techs = ('git',) + ('aws',) -> ('git', 'aws')
techs = ds_techs + dev_techs -> ('sql', 'pandas', 'git', 'aws')

Question: 502

Does a user-defined function need to include the return keyword?

A. No
B. Yes

Answer: A

Explanation:
Example: def hello():
print('Welcome')
https://docs.python.org/3/tutorial/controlflow.html#defining-functions

Question: 503

How can you create multiline comments in Python? (select 3)

A. A multilinecommentcan be placed between * and *
B. A multilinecommentcan be placed between the string"""(triplequotation marks)
C. A multilinecommentcan be placed between /* and */
D. We can create a multiline comment from multiple single-linecommentsbyinserting the # sign at the beginning of each line
E. A multiline comment can be placed between the string ''' (triple apostrophe)
F. A multiline comment can be placed between -- and -

Answer: B,D,E **Explanation:** Example: '''This is a multiline comment''' """This is a multiline comment"""

Question: 504

What does LEGB stand for?

A. The LEGB abbreviation consists of the first letters of the three Python scope names: Local, Enclosing, Global.
B. The LEGB abbreviation consists of the first letters of the Python built-in classes: local, enclosing, global, builtin.
C. The LEGB abbreviation consists of the first letters of all Python scope names: Local, Enclosing, Global, Built-in.

Answer: C

Question: 505

The following two lists are given: tech_names = ['python', 'django', 'html'] exps = ['senior', 'mid', 'junior'] We want to iterate through these lists simultaneously. What built-in function will you use for this purpose?

A. map()
B. zip()
C. zip_longest()
D. filter()

E. reduce()

Answer: B

Explanation:
zip(*iterables, strict=False) - iterate over several iterables in parallel, producing tuples with an item from each one.
https://docs.python.org/3/library/functions.html#zip

Question: 506

An operator able to check whether two values are not equal in Python is coded as:

A. not ==
B. !=
C. ==
D. <>

Answer: B

Explanation:
!= - returns a Boolean stating whether two expressions are not equal.

Question: 507

Can a list be a key in a dictionary?

A. No, lists are mutable and dictionary keys must be immutable objects.
B. Yes

Answer: A

Question: 508

What exception do you expect if your code has a syntax error?

A. ModuleNotFoundError
C. SyntaxError
D. NameError
E. RuntimeError
F. IndexError

Answer: B

Explanation:
SyntaxError is raised when the parser encounters a syntax error.
https://docs.python.org/3/library/exceptions.html#SyntaxError

Question: 509

What function can you use to check the type of an object?

A. dir()
B. help()
C. object()
D. type()
E. typeof()

Answer: D

Explanation:
class type(name, bases, dict, **kwds) - with one argument, return the type of an object.
https://docs.python.org/3/library/functions.html#type

Question: 510

You have to create your own exception. What class will you use to inherit from?

A. Exception
B. Warning
C. RuntimeError
D. Error
E. AssertionError

Answer: A

Explanation:
Exception - all built-in, non-system-exiting exceptions are derived from this class. All user-defined exceptions should also be derived from this class.
https://docs.python.org/3/library/exceptions.html#Exception

Question: 511

What is the output of the following code snippet if the user enters 8 and 4 respectively?

```
var1 =int(input())
var2 =int(input())

var1 = var1 % var2
var2 =var2 // var1
print(var2)
```

A. 8
B. 4
C. 0
D. This code will raise the TypeError exception.
E. This code will raise the ZeroDivisionError exception.

Answer: E

Explanation:

Steps:
var1 = int(input()) -> 8
var2 = int(input()) -> 4
var1 = var1 % var2 -> 8 % 4 -> 0
var2 = var2 // var1 -> 4 // 0 -> It will raise the ZeroDivisionError exception.

Question: 512

What is the output of the following print function?
print('sport', 'running', 'trenning', sep=',')

A. sportrunningtrenning,
B. sport,running,trenning
C. sportrunningtrenning
D. sport running trenning
E. sport running trenning,

Answer: B

Explanation:

print(*objects, sep=' ', end='
', file=sys.stdout, flush=False) - print objects to the text stream file, separated by sep and followed by end. sep, end, file, and flush, if present, must be given as keyword arguments.
All non-keyword arguments are converted to strings like str() does and written to the stream, separated by sep and followed by end. Both sep and end must be strings; they can also be None, which means to use the default values. If no objects are given, print() will just write end.
https://docs.python.org/3/library/functions.html#print

Question: 513

The 0x prefix means that the number after it is denoted as...

A. hexadecimal.
B. decimal.
C. binary.
D. octal.

Answer: A

Question: 514

The following list is given:
artists = ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC'] What will be the result of the following operation?

A. artists[::-1]
B. ['Sting', 'Phil Collins', 'The Police', 'Queen', 'AC/DC']
C. 'AC/DC'
D. ['AC/DC', 'Queen', 'The Police', 'Phil Collins', 'Sting']

E. ['Phil Collins', 'The Police', 'Queen', 'AC/DC']
F. ['Sting', 'Phil Collins', 'The Police', 'Queen']
Answer: D

Question: 515

Which Python keyword should you use to create a generator?

A. yield
B. return

C. iter
D. next

Answer: A

Explanation:

yield_stmt ::= yield_expression

Yield expressions and statements are only used when defining a generator function, and are only used in the body of the generator function. Using yield in a function definition is sufficient to cause that definition to create a generator function instead of a normal function.

https://docs.python.org/3/reference/simple_stmts.html#the-yield-statement

Question: 516

Which of the following variable names are not allowed? (select 2)

A. True
B. AND
C. and
D. TRUE
E. true

Answer: A,C

Explanation:

True and and are reserved Python keywords.

Question: 517

What will be the result of the following operation?

A. 'python' * False
B. 'python'
C. ''
D. The operation will not be performed. Objects like str and bool cannot be multiplied.
E. False

Answer: C

Question: 518

What is the output of the following snippet? x = 10 ** 2 + 10 / 2 + 10 // 5 + 13 % 5 print(x)

A. 110.0
B. 30.0
C. 107.0
D. The given expression is incorrect, it will raise SyntaxError exception.

Answer: A

Explanation:

Steps:

x = 10 ** 2 + 10 / 2 + 10 // 5 + 13 % 5 -> 100 + 5.0 + 2 + 3 -> 110.0

Question: 519

The following list is given: numbers = list(range(20)) What will be the result of the following? numbers[::2]

A. [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
B. [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

C. [0, 1]
D. [2, 4, 6, 8, 10, 12, 14, 16, 18]
Answer: B
Explanation:
Steps:
list(range(20)) -> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
numbers[::2] -> [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

Question: 520

What is the expected behavior of the following program?

A. numbers = (1, 10, 100, 1000)
B. numbers.index(0)
C. This code will cause the NameError exception.
D. This code will cause the AttributeError exception.
E. This code will cause the ValueError exception.
F. This code will output None to the console.
G. This code will output 1 to the console.
Answer: E
Explanation:
tuple.index(self, value, start=0, stop=9223372036854775807, /) - returna first index of value. Raises ValueError if the value is not present.

Question: 521

What function will you use to get input from the user?

A. print()
B. read()
C. input()
D. stdin()
Answer: C
Explanation:
input([prompt]) - if the prompt argument is present, it is written to standard output without a trailing newline. The function then reads a line from input, converts it to a string (stripping a trailing newline), and returns that.
https://docs.python.org/3/library/functions.html#input

Question: 522

Select all true statements about Python operators. (select 2)

A. The right argument of the // operator can be zero.
B. The == operator is an example of assignment operator.
C. The ** operator returns the value of a numeric expression raised to a specified power.
D. The right argument of the % operator cannot be zero.
Answer: C,D

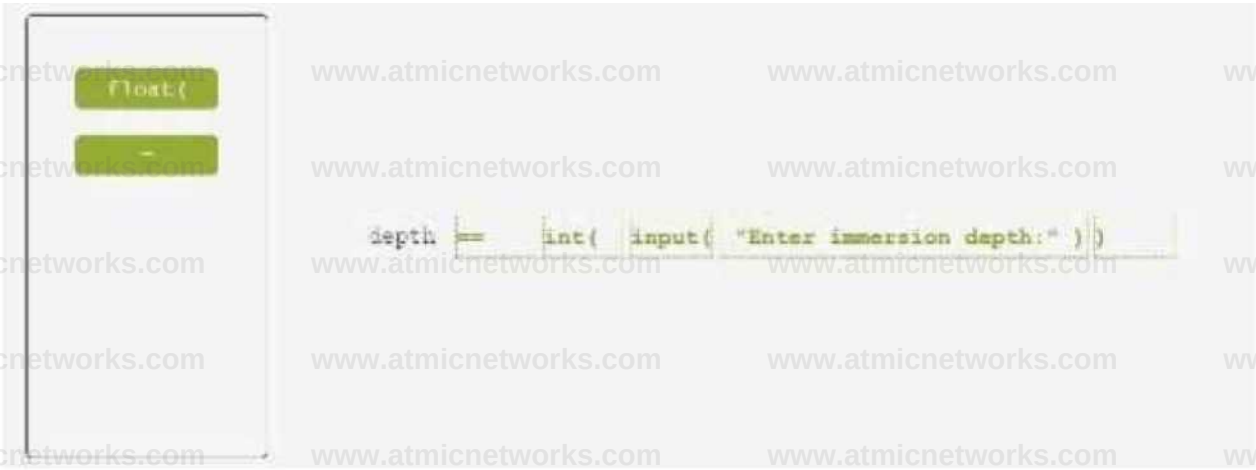
Question: 523

DRAG DROP

Insert the code boxes in the correct positions in order to build a line of code which asks the user for an integer value and assigns it to the depth variable.

(Note: some code boxes will not be used.)

Answer:



One possible way to insert the code boxes in the correct positions in order to build a line of code which asks the user for an integer value and assigns it to the depth variable is:

depth = int(input("Enter the immersion depth: "))

This line of code uses the input function to prompt the user for a string value, and then uses the int function to convert that string value into an integer number. The result is then assigned to the variable depth.

You can find more information about the input and int functions in Python in the following references:

[Python input()
Function]

[Python int()

Function]

Question: 524

A set of rules which defines the ways in which words can be coupled in sentences is called:

A. lexis

B. syntax

C. semantics

D. dictionary

Answer: C,D

Explanation:

Syntax is the branch of linguistics that studies the structure and rules of sentences in natural languages. Lexis is the vocabulary of a language. Semantics is the study of meaning in language. A dictionary is a collection of words and their definitions, synonyms, pronunciations, etc.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 525

DRAG DROP

Arrange the binary numeric operators in the order which reflects their priorities, where the top-most position has the highest priority and the bottom-most position has the lowest priority.

Answer:

Explanation:

*

The correct order of the binary numeric operators in Python according to their priorities is:

Exponentiation (**)
Multiplication (*) and Division (/, //, %)

Addition (+) and Subtraction (-)

This order follows the standard mathematical convention of operator precedence, which can be remembered by the acronym PEMDAS (Parentheses, Exponents, Multiplication/Division, Addition/Subtraction). Operators with higher precedence are evaluated before those with lower precedence, but operators with the same precedence are evaluated from left to right. Parentheses can be used to change the order of evaluation by grouping expressions.

For example, in the expression 2 + 3 * 4 ** 2, the exponentiation operator (**) has the highest priority, so it is evaluated first, resulting in 2 + 3 * 16. Then, the multiplication operator (*) has the next highest priority, so it is evaluated next, resulting in 2 + 48. Finally, the addition operator (+) has the lowest priority, so it is evaluated last, resulting in 50.

You can find more information about the operator precedence in Python in the following references:

6. Expressions — Python 3.11.5 documentation

Precedence and Associativity of Operators in Python - Programiz
Python Operator Priority or Precedence Examples Tutorial

Question: 526

Which of the following expressions evaluate to a non-zero result? (Select two answers.)

A. 2 ** 3 / A - 2

B. 4 / 2 * * 3 - 2

C. 1 * * 3 / 4 - 1

D. 1 * 4 // 2 ** 3

Answer: AB

Explanation:

In Python, the ** operator is used for exponentiation, the / operator is used for floating-point division, and the // operator is used for integer division. The order of operations is parentheses, exponentiation, multiplication/division, and addition/subtraction. Therefore, the expressions can be

evaluated as follows:

A) 2 ** 3 / A - 2 = 8 / A - 2 (assuming A is a variable that is not zero or undefined) B. 4 / 2 * * 3 - 2 = 4 / 8 - 2 = 0.5 - 2 = -1.5 C. 1 * * 3 / 4 - 1 = 1 / 4 - 1 = 0.25 - 1 = -0.75 D. 1 * 4 // 2 ** 3 = 4 // 8 = 0

Only expressions A and B evaluate to non-zero results.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 527

DRAG DROP

Insert the code boxes in the correct positions in order to build a line of code which asks the user for a float value and assigns it to the mass variable.

(Note: some code boxes will not be used.)

Answer:

Explanation:

Hump = (float i(input i ("Enter

One possible way to insert the code boxes in the correct positions in order to build a line of code that asks the user for a float value and assigns it to the mass variable is:

mass = float(input("Enter the mass: "))

This line of code uses the input function to prompt the user for a string value, and then uses the float function to convert that string value into a floating-point number. The result is then assigned to the variable mass.

You can find more information about the input and float functions in Python in the following references:

[Python input() Function]
[Python float() Function]

Question: 528

Python Is an example of which programming language category?

A. interpreted

B. assembly

C. compiled

D. machine

Answer: A

Explanation:

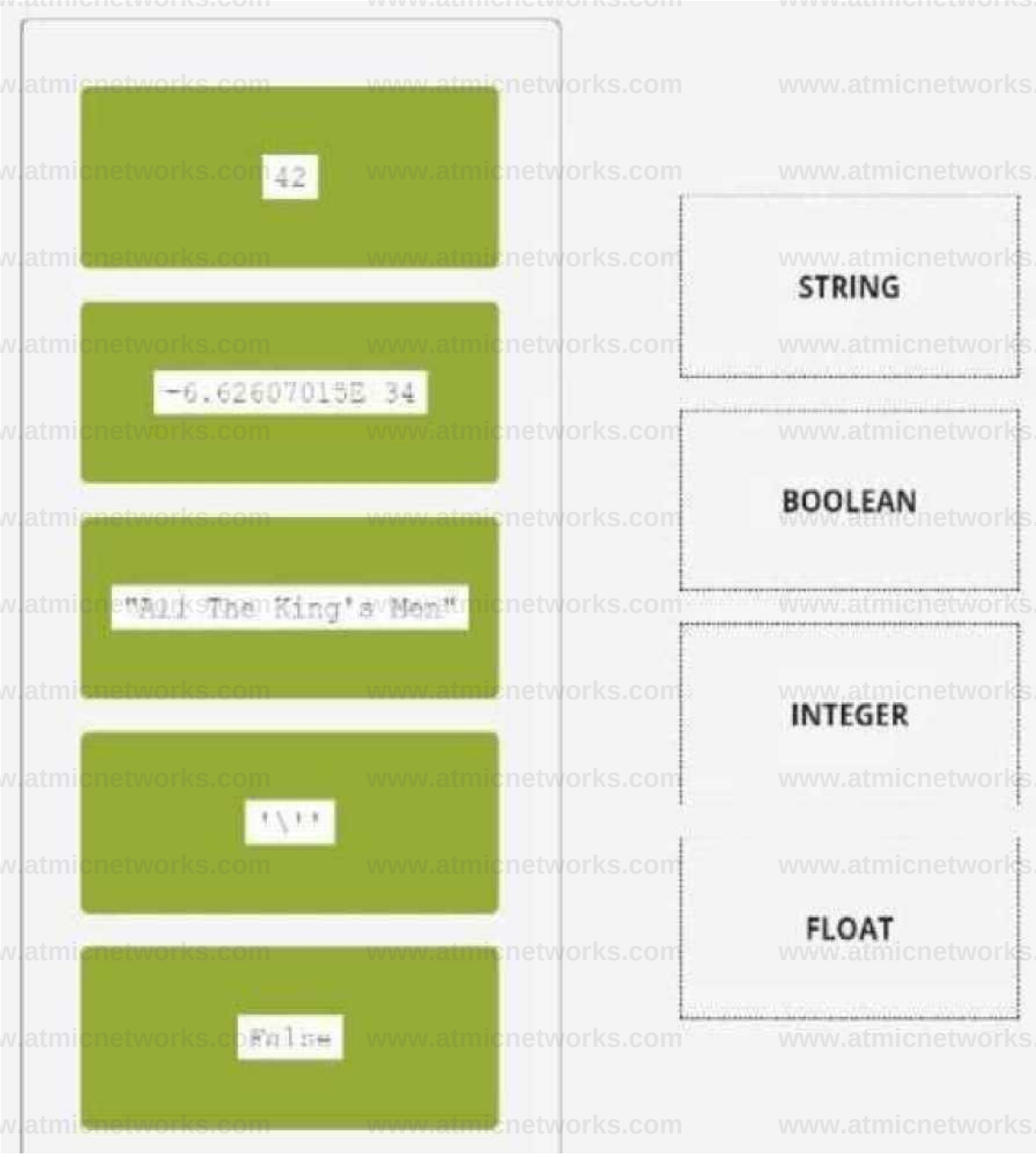
Python is an interpreted programming language, which means that the source code is translated into executable code by an interpreter at runtime, rather than by a compiler beforehand. Interpreted languages are more flexible and portable than compiled languages, but they are also slower and less efficient. Assembly and machine languages are low-level languages that are directly executed by the hardware, while compiled languages are high-level languages that are translated into machine code by a compiler before execution.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 529

DRAG DROP

Drag and drop the literals to match their data type names.



Answer:

Explanation:

One possible way to drag and drop the literals to match their data type names is:

STRING: “All The King’s Men”

BOOLEAN: False

INTEGER: 42

FLOAT: -6.62607015E-34

A literal is a value that is written exactly as it is meant to be interpreted by the Python interpreter. A data type is a category of values that share some common characteristics or operations. Python has four basic data types: string, boolean, integer, and float.

A string is a sequence of characters enclosed by either single or double quotes. A string can represent text, symbols, or any other information that can be displayed as text. For example, “All The King’s Men” is a string literal that represents the title of a novel.

A boolean is a logical value that can be either True or False. A boolean can represent the result of a comparison, a condition, or a logical operation. For example, False is a boolean literal that represents the opposite of True.

An integer is a whole number that can be positive, negative, or zero. An integer can represent a count, an index, or any other quantity that does not require fractions or decimals. For example, 42 is an integer literal that represents the answer to life, the universe, and everything.

A float is a number that can have a fractional part after the decimal point. A float can represent a measurement, a ratio, or any other quantity that requires precision or approximation. For example, - 6.62607015E-34 is a float literal that represents the Planck constant in scientific notation.

You can find more information about the literals and data types in Python in the following references:

[Python Data Types]

[Python Literals]

[Python Basic Syntax]

Question: 530

How many hashes (+) does the code output to the screen?

A. one

B. zero (the code outputs nothing)

C. five

D. three

Answer: C

Explanation:

The code snippet that you have sent is a loop that checks if a variable “floor” is less than or equal to 0 and prints a string accordingly. The code is as follows:

floor = 5 while floor > 0: print(“+”) floor = floor - 1

The code starts with assigning the value 5 to the variable “floor”. Then, it enters a while loop that repeats as long as the condition “floor > 0” is true. Inside the loop, the code prints a “+” symbol to the screen, and then subtracts 1 from the value of “floor”. The loop ends when “floor” becomes 0 or negative, and the code exits.

The code outputs five “+” symbols to the screen, one for each iteration of the loop. Therefore, the correct answer is C. five.

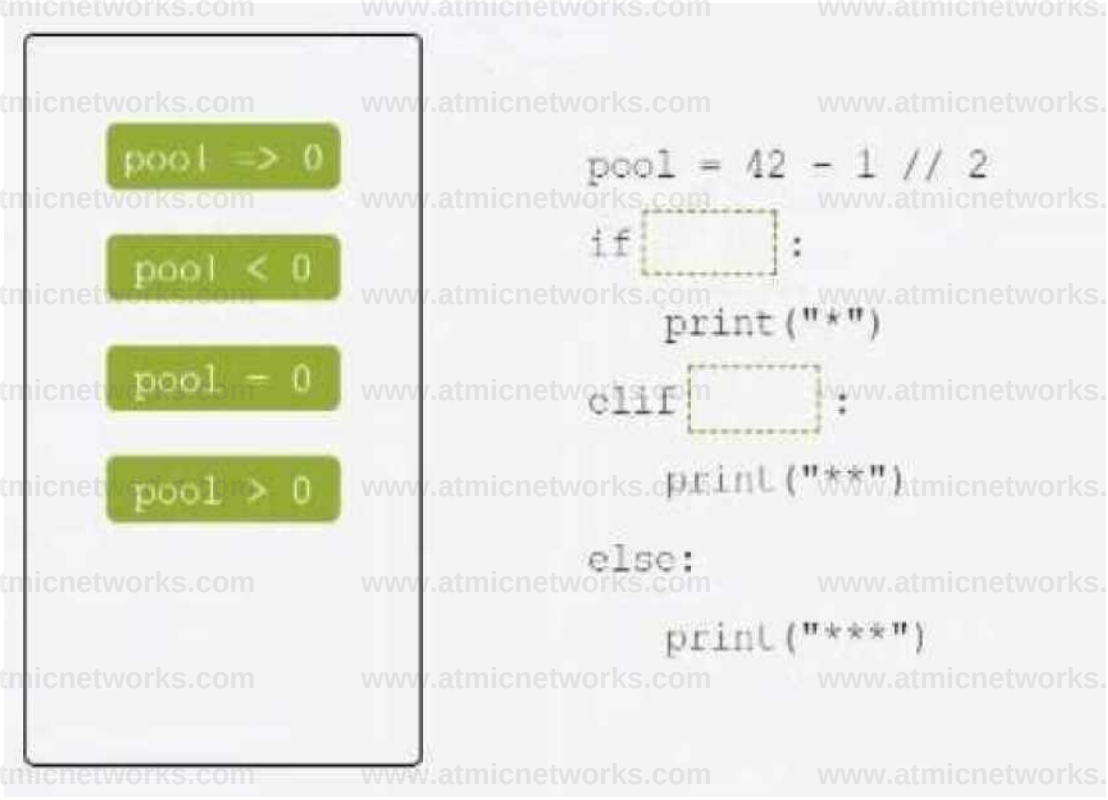
Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 531

DRAG DROP

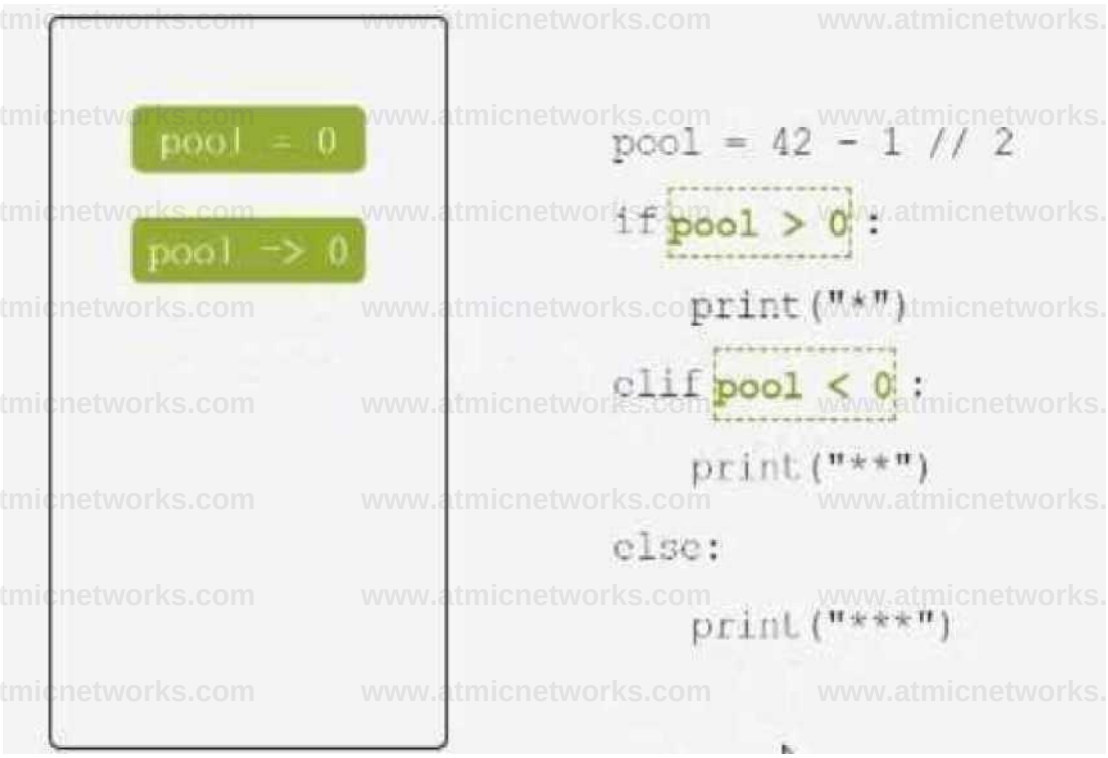
Drag and drop the conditional expressions to obtain a code which outputs * to the screen.

(Note: some code boxes will not be used.)



Answer:

Explanation:



One possible way to drag and drop the conditional expressions to obtain a code which outputs * to the screen is:

```
if pool > 0:
  print("*")
elif pool < 0:
  print("**")
else:
  print("***")
```

This code uses the if, elif, and else keywords to create a conditional statement that checks the value of the variable pool. Depending on whether the value is greater than, less than, or equal to zero, the code will print a different pattern of asterisks to the screen. The print function is used to display the output. The code is indented to show the blocks of code that belong to each condition. The code will output * if the value of pool is positive, ** if the value of pool is negative, and *** if the value of pool is zero.

You can find more information about the conditional statements and the print function in Python in the following references:

[Python If … Else]

[Python Print Function]

[Python Basic Syntax]

Question: 532

What happens when the user runs the following code?

```
Lot al - 0
for i in ranged):
    If 2 * 1 < 4: total +- 1
    total += 1
pr i it (tota l)
```

A. The code outputs 3.

B. The code outputs 2.

C. The code enters an infinite loop.

D. The code outputs 1.

Answer: B

Explanation:

The code snippet that you have sent is calculating the value of a variable “total” based on the values in the range of 0 to 3. The code is as follows:

total = 0 for i in range(0, 3): if i % 2 == 0: total = total + 1 else: total = total + 2 print(total)

The code starts with assigning the value 0 to the variable “total”. Then, it enters a for loop that iterates over the values 0, 1, and 2 (the range function excludes the upper bound). Inside the loop, the code checks if the current value of “i” is even or odd using the modulo operator (%). If “i” is even, the code adds 1 to the value of “total”. If “i” is odd, the code adds 2 to the value of “total”. The loop ends when “i” reaches 3, and the code prints the final value of “total” to the screen.

The code outputs 2 to the screen, because the value of “total” changes as follows:

When i = 0, total = 0 + 1 = 1

When i = 1, total = 1 + 2 = 3

When i = 2, total = 3 + 1 = 4

When i = 3, the loop ends and total = 4 is printed

Therefore, the correct answer is B. The code outputs 2.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 533

What is the expected output of the following code?

```
counter = 84 // 2
if counter < 0:
    print("*")
elif counter >= 42:
    print("**")
else:
    print("***")
```

A. The code produces no output.

B. * * *

C. * *

D. *

Answer: C

Explanation:

The code snippet that you have sent is a conditional statement that checks if a variable “counter” is less than 0, greater than or equal to 42, or neither. The code is as follows:

if counter < 0: print(“”) elif counter >= 42: print(””) else: print(””)

The code starts with checking if the value of “counter” is less than 0. If yes, it prints a single asterisk () to the screen and exits the statement. If no, it checks if the value of “counter” is greater than or equal to 42. If yes, it prints three asterisks () to the screen and exits the statement. If no, it prints two asterisks () to the screen and exits the statement.

The expected output of the code depends on the value of “counter”. If the value of “counter” is 10, as shown in the image, the code will print two asterisks (**) to the screen, because 10 is neither less than 0 nor greater than or equal to 42. Therefore, the correct answer is C. * *

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 534

DRAG DROP

Arrange the code boxes in the correct positions in order to obtain a loop which executes its body with the level variable going through values 5, 1, and 1 (in the same order).

Answer:

Explanation:

Question: 535

What happens when the user runs the following code?

```
whil~ sp^ed

    if speed > IC
        Vint' nue
    print("'"
```

A. The program outputs three asterisks (***)to the screen.

B. The program outputs one asterisk (*) to the screen.

C. The program outputs five asterisks (*****) to the screen.

D. The program enters an infinite loop.

Answer: D

Explanation:

The code snippet that you have sent is a while loop with an if statement and a print statement inside it. The code is as follows:

while True: if counter < 0: print(“”) else: print(”**”)

The code starts with entering a while loop that repeats indefinitely, because the condition “True” is always true. Inside the loop, the code checks if the value of “counter” is less than 0. If yes, it prints a single asterisk () to the screen. If no, it prints three asterisks (**) to the screen. However, the code does not change the value of “counter” inside the loop, so the same condition is checked over and over again. The loop never ends, and the code enters an infinite loop.

The program outputs either one asterisk () or three asterisks (**) to the screen repeatedly, depending on the initial value of “counter”. Therefore, the correct answer is D. The program enters an infinite loop.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 536

What is the expected output of the following code?

```
equals = 0
for i in range(2):
    for j in range(2):
        if i == j:
            equals += 1
else:
    equals += 1
print(equals)
```

A. The code outputs nothing.

B. 3

C. 1

D. 4

Answer: C

Explanation:

The code snippet that you have sent is checking if two numbers are equal and printing the result. The code is as follows:

num1 = 1 num2 = 2 if num1 == num2: print(4) else: print(1)

The code starts with assigning the values 1 and 2 to the variables “num1” and “num2” respectively. Then, it enters an if statement that compares the values of “num1” and “num2” using the equality operator (==). If the values are equal, the code prints 4 to the screen. If the values are not equal, the code prints 1 to the screen.

The expected output of the code is 1, because the values of “num1” and “num2” are not equal. Therefore, the correct answer is C. 1.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 537

DRAG DROP

Arrange the code boxes in the correct positions to form a conditional instruction which guarantees that a certain statement is executed when the speed variable is less than 50.0.

speed : < if 50.0

Answer:

Explanation:

if speed < 50.0

One possible way to arrange the code boxes in the correct positions to form a conditional instruction which guarantees that a certain statement is executed when the speed variable is less than 50.0 is:

```
if speed < 50.0:
    print("The speed is low.")
```

This code uses the if keyword to create a conditional statement that checks the value of the variable speed. If the value is less than 50.0, then the code will print “The speed is low.” to the screen. The print function is used to display the output. The code is indented to show the block of code that belongs to the if condition.

You can find more information about the if statement and the print function in Python in the following references:

Python If … Else

Python Print Function

Question: 538

What is the expected output of the following code?

```
colleu iun - I)
col lection. App-no (1)
collection. In.'; iLC?, 2) duplicate - collection
 1' 11 I 1 • \i          . J ! ~     ( )
print (1 .ri (collection) - l.n(duplicate))
```

A. 5

B. 4

C. 6

D. The code raises an exception and outputs nothing.

Answer: D

Explanation:

The code snippet that you have sent is trying to print the combined length of two lists, "collection" and "duplicate". The code is as follows:

collection = [] collection.append(1) collection.insert(0, 2) duplicate = collection duplicate.append(3) print(len(collection) + len(duplicate))

The code starts with creating an empty list called "collection" and appending the number 1 to it. The list now contains [1]. Then, the code inserts the number 2 at the beginning of the list. The list now contains [2, 1]. Then, the code creates a new list called "duplicate" and assigns it the value of "collection". However, this does not create a copy of the list, but rather a reference to the same list object. Therefore, any changes made to "duplicate" will also affect "collection", and vice versa. Then, the code appends the number 3 to "duplicate". The list now contains [2, 1, 3], and so does "collection". Finally, the code tries to print the sum of the lengths of "collection" and "duplicate".

However, this causes an exception, because the len function expects a single argument, not two. The code does not handle the exception, and therefore outputs nothing.

The expected output of the code is nothing, because the code raises an exception and terminates.

Therefore, the correct answer is D. The code raises an exception and outputs nothing.

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 539

Assuming that the following assignment has been successfully executed:

My_list – [1, 1, 2, 3]

Select the expressions which will not raise any exception.

(Select two expressions.)

A. my_list[-10]

B. my_list|my_Li1st | 3| I

C. my list [6]

D. my_List- [0:1]

Answer: B, D

Explanation:

The code snippet that you have sent is assigning a list of four numbers to a variable called "my_list". The code is as follows:

my_list = [1, 1, 2, 3]

The code creates a list object that contains the elements 1, 1, 2, and 3, and assigns it to the variable "my_list". The list can be accessed by using the variable name or by using the index of the elements. The index starts from 0 for the first element and goes up to the length of the list minus one for the

last element. The index can also be negative, in which case it counts from the end of the list. For example, my_list[0] returns 1, and my_list[-1] returns 3.

The code also allows some operations on the list, such as slicing, concatenation, repetition, and membership. Slicing is used to get a sublist of the original list by specifying the start and end index. For example, my_list[1:3] returns [1, 2]. Concatenation is used to join two lists together by using the + operator. For example, my_list + [4, 5] returns [1, 1, 2, 3, 4, 5]. Repetition is used to create a new list by repeating the original list a number of times by using the * operator. For example, my_list * 2 returns [1, 1, 2, 3, 1, 1, 2, 3]. Membership is used to check if an element is present in the list by using the in operator. For example, 2 in my_list returns True, and 4 in my_list returns False.

The expressions that you have given are trying to access or manipulate the list in different ways. Some of them are valid, and some of them are invalid and will raise an exception. An exception is an error that occurs when the code cannot be executed properly. The expressions are as follows:

A) my_list[-10]: This expression is trying to access the element at the index -10 of the list. However, the list only has four elements, so the index -10 is out of range. This will raise an IndexError exception and output nothing.

B) my_list|my_Li1st | 3| I: This expression is trying to perform a bitwise OR operation on the list and some other operands. The bitwise OR operation is used to compare the binary representation of two numbers and return a new number that has a 1 in each bit position where either number has a 1. For example, 3 | 1 returns 3, because 3 in binary is 11 and 1 in binary is 01, and 11 | 01 is 11. However, the bitwise OR operation cannot be applied to a list, because a list is not a number. This will raise a TypeError exception and output nothing.

C) my list [6]: This expression is trying to access the element at the index 6 of the list. However, the list only has four

elements, so the index 6 is out of range. This will raise an IndexError exception and output nothing.

D) my_List- [0:1]: This expression is trying to perform a subtraction operation on the list and a sublist. The subtraction operation is used to subtract one number from another and return the difference. For example, 3 - 1 returns 2. However, the subtraction operation cannot be applied to a list, because a list is not a number. This will raise a TypeError exception and output nothing.

Only two expressions will not raise any exception. They are:

B) my_list|my_Li1st | 3| I: This expression is not a valid Python code, but it is not an expression that tries to access or manipulate the list. It is just a string of characters that has no meaning. Therefore, it will not raise any exception, but it will also not output anything.

E) my_List- [0:1]: This expression is a valid Python code that uses the slicing operation to get a sublist of the list. The slicing operation does not raise any exception, even if the start or end index is out of range. It will just return an empty list or the closest possible sublist. For example, my_list[0:10] returns [1, 1, 2, 3], and my_list[10:20] returns []. The expression my_List- [0:1] returns the sublist of the list from the index 0 to the index 1, excluding the end index. Therefore, it returns [1]. This

expression will not raise any exception, and it will output [1].

Therefore, the correct answers are B. my_list|my_Li1st | 3| I and D. my_List- [0:1].

Reference: [Python Institute - Entry-Level Python Programmer Certification]

Question: 540

What is true about tuples? (Select two answers.)

A. Tuples are immutable, which means that their contents cannot be changed during their lifetime.

B. The len { } function cannot be applied to tuples.

C. An empty tuple is written as { } .

D. Tuples can be indexed and sliced like lists.

Answer: A, D

Explanation:

Tuples are one of the built-in data types in Python that are used to store collections of data. Tuples have some

characteristics that distinguish them from other data types, such as lists, sets, and dictionaries. Some of these characteristics are:

Tuples are immutable, which means that their contents cannot be changed during their lifetime. Once a tuple is created, it cannot be modified, added, or removed. This makes tuples more stable and reliable than mutable data types. However, this also means that tuples are less flexible and dynamic than mutable data types. For example, if you want to change an element in a tuple, you have to create a new tuple with the modified element and assign it to the same variable12

Tuples are ordered, which means that the items in a tuple have a defined order and can be accessed by using their index. The index of a tuple starts from 0 for the first item and goes up to the length of the tuple minus one for the last item. The index can also be negative, in which case it counts from the end of the tuple. For example, if you have a tuple t = ("a", "b", "c"), then t[0] returns "a", and t[-1] returns "c"12

Tuples can be indexed and sliced like lists, which means that you can get a single item or a sublist of a tuple by using square brackets and specifying the start and end index. For example, if you have a

tuple t = ("a", "b", "c", "d", "e"), then t[2] returns "c", and t[1:4] returns ("b", "c", "d"). Slicing does not raise any exception, even if the start or end index is out of range. It will just return an empty tuple or the closest possible sublist12

Tuples can contain any data type, such as strings, numbers, booleans, lists, sets, dictionaries, or even other tuples. Tuples can also have duplicate values, which means that the same item can appear more than once in a tuple. For example, you can have a tuple t = (1, 2, 3, 1, 2), which contains two 1s and two 2s12

Tuples are written with round brackets, which means that you have to enclose the items in a tuple with parentheses. For example, you can create a tuple t = ("a", "b", "c") by using round brackets. However, you can also create a tuple without using round brackets, by just separating the items with commas. For example, you can create the same tuple t = "a", "b", "c" by using commas. This is called tuple packing, and it allows you to assign multiple values to a single variable12

The len() function can be applied to tuples, which means that you can get the number of items in a tuple by using the len() function. For example, if you have a tuple t = ("a", "b", "c"), then len(t) returns 312

An empty tuple is written as (), which means that you have to use an empty pair of parentheses to create a tuple with no items. For example, you can create an empty tuple t = () by using empty parentheses. However, if you want to create a tuple with only one item, you have to add a comma after the item, otherwise Python will not recognize it as a tuple. For example, you can create a tuple with one item t = ("a",) by using a comma12

Therefore, the correct answers are A. Tuples are immutable, which means that their contents cannot be changed during their lifetime. and D. Tuples can be indexed and sliced like lists.

Reference: Python Tuples - W3SchoolsTuples in Python - GeeksforGeeks

Question: 541

DRAG DROP

Assuming that the phonc_dir dictionary contains namemumber pairs, arrange the code boxes to create a valid line of code which retrieves Martin Eden's phone number, and assigns it to the number variable.

number "Martin Eden* phonedir

Answer:

Explanation:

number = phone_dir ["Martin Eden"]

number = phone_dir["Martin Eden"]

This code uses the square brackets notation to access the value associated with the key “Martin Eden” in the phone_dir dictionary. The value is then assigned to the variable number. A dictionary is a data structure that stores key-value pairs, where each key is unique and can be used to retrieve its corresponding value. You can find more information about dictionaries in Python in the following references:

[Python Dictionaries - W3Schools]

[Python Dictionary (With Examples) - Programiz]

[5.5. Dictionaries — How to Think Like a Computer Scientist …]

Question: 542

Assuming that the following assignment has been successfully executed:

the list - I, 1.]

Which of the following expressions evaluate to True? (Select two expressions.)

A. the_List.index {"1"} in the_list

B. 1.1 in the_list |1:3 |

C. len (the list [0:2]} <3

D. the_list. index {'1'} -- 0

Answer: CD

Explanation:

The code snippet that you have sent is assigning a list of four values to a variable called “the_list”. The code is as follows:

the_list = [‘1’, 1, 1, 1]

The code creates a list object that contains the values ‘1’, 1, 1, and 1, and assigns it to the variable “the_list”. The list can be accessed by using the variable name or by using the index of the values. The index starts from 0 for the first value and goes up to the length of the list minus one for the last value. The index can also be negative, in which case it counts from the end of the list. For example, the_list[0] returns ‘1’, and the_list[-1] returns 1.

The expressions that you have given are trying to evaluate some conditions on the list and return a boolean value, either True or False. Some of them are valid, and some of them are invalid and will raise an exception. An exception is an error that occurs when the code cannot be executed properly. The expressions are as follows:

A) the_List.index {“1”} in the_list: This expression is trying to check if the index of the value ‘1’ in the list is also a value in the list. However, this expression is invalid, because it uses curly brackets instead of parentheses to call the index method. The index method is used to return the first occurrence of a value in a list. For example, the_list.index(‘1’) returns 0, because ‘1’ is the first value in the list. However, the_list.index {“1”} will raise a SyntaxError exception and output nothing.

B) 1.1 in the_list |1:3 |: This expression is trying to check if the value 1.1 is present in a sublist of the list. However, this expression is invalid, because it uses a vertical bar instead of a colon to specify the start and end index of the sublist. The sublist is obtained by using the slicing operation, which uses square brackets and a colon to get a part of the list. For example, the_list[1:3] returns [1, 1], which is the sublist of the list from the index 1 to the index 3, excluding the end index. However, the_list |1:3 | will raise a SyntaxError exception and output nothing.

C) len (the list [0:2]} <3: This expression is trying to check if the length of a sublist of the list is less than 3. This expression is valid, because it uses the len function and the slicing operation correctly. The len function is used to return the number of values in a list or a sublist. For example, len(the_list) returns 4, because the list has four values. The slicing operation is used to get a part of the list by using square brackets and a colon. For example, the_list[0:2] returns [‘1’, 1], which is the sublist of the list from the index 0 to the index 2, excluding the end index. The expression len (the list [0:2]} <3

returns True, because the length of the sublist [‘1’, 1] is 2, which is less than 3.

D) the_list. index {‘1’} – 0: This expression is trying to check if the index of the value ‘1’ in the list is equal to 0. This expression is valid, because it uses the index method and the equality operator correctly. The index method is used to return the first occurrence of a value in a list. For example, the_list.index(‘1’) returns 0, because ‘1’ is the first value in the list. The equality operator is used to compare two values and return True if they are equal, or False if they are not. For example, 0 == 0 returns True, and 0 == 1 returns False. The expression the_list. index {‘1’} – 0 returns True, because the index of ‘1’ in the list is 0, and 0 is equal to 0.

Therefore, the correct answers are C. len (the list [0:2]} <3 and D. the_list. index {‘1’} – 0.

Reference: Python List Methods - W3Schools5. Data Structures — Python 3.11.5 documentationList methods in Python - GeeksforGeeks

Question: 543

What is the expected output of the following code?

```
l          |"(          ":     . -, " ii": l.'-"\ ” l - p ":                    ■

lor value in menu:
        print(str(value)[0], end-"")
```

A. The code is erroneous and cannot be run.

B. ppt

C. 213

D. pizzapastafolpetti

Answer: B

Explanation:

The code snippet that you have sent is using the slicing operation to get parts of a string and concatenate them together. The code is as follows:

pizza = “pizza” pasta = “pasta” folpetti = “folpetti” print(pizza[0] + pasta[0] + folpetti[0])

The code starts with assigning the strings “pizza”, “pasta”, and “folpetti” to the variables pizza, pasta, and folpetti respectively. Then, it uses the print function to display the result of concatenating the first characters of each string. The first character of a string can be accessed by using the index 0 inside square brackets. For example, pizza[0] returns “p”. The concatenation operation is used to join two or more strings together by using the + operator. For example, “a” + “b” returns “ab”. The code prints the result of pizza[0] + pasta[0] + folpetti[0], which is “p” + “p” + “f”, which is “ppt”.

The expected output of the code is ppt, because the code prints the first characters of each string.

Therefore, the correct answer is B. ppt.

Reference: Python String Slicing - W3SchoolsPython String Concatenation - W3Schools

Question: 544

What is the expected result of the following code?

```
rales - (1.2, 1.4, 1.0)
new - rates^:]
for rate in rates[-2:1: new +- (rate,)
pri it. (len (new))
```

A. 5

B. 2

C. 1

D. The code will cause an unhandled

Answer: D

Explanation:

The code snippet that you have sent is trying to use a list comprehension to create a new list from an existing list. The code is as follows:

my_list = [1, 2, 3, 4, 5] new_list = [x for x in my_list if x > 5]

The code starts with creating a list called “my_list” that contains the numbers 1, 2, 3, 4, and 5. Then, it tries to create a new list called “new_list” by using a list comprehension. A list comprehension is a concise way of creating a new list from an existing list by applying some expression or condition to each element. The syntax of a list comprehension is:

new_list = [expression for element in old_list if condition]

The expression is the value that will be added to the new list, which can be the same as the element or a modified version of it. The element is the variable that takes each value from the old list. The condition is an optional filter that determines which elements will be included in the new list. For example, the following list comprehension creates a new list that contains the squares of the even numbers from the old list:

old_list = [1, 2, 3, 4, 5, 6] new_list = [x ** 2 for x in old_list if x % 2 == 0]

new_list = [4, 16, 36]

The code that you have sent is trying to create a new list that contains the elements from the old list that are greater than 5. However, there is a problem with this code. The problem is that none of the elements in the old list are greater than 5, so the condition is always false. This means that the new list will be empty, and the expression will never be evaluated. However, the expression is not valid, because it uses the variable x without defining it. This will cause a NameError exception, which is an error that occurs when a variable name is not found in the current scope. The code does not handle the exception, and therefore it will terminate with an error message.

The expected result of the code is an unhandled exception, because the code tries to use an undefined variable in an expression that is never executed. Therefore, the correct answer is D. The code will cause an unhandled exception.

Reference: Python - List Comprehension - W3SchoolsPython - List Comprehension - GeeksforGeeksPython Exceptions: An Introduction – Real Python

Question: 545

DRAG DROP

Drag and drop the code boxes in order to build a program which prints Unavailable to the screen.

(Note: one code box will not be used.)

prices - { -pizza": 3.49)

try:

chaiqe - pt 1cesI"■ MIzone print("Charged")

priritrnnavai I able-)

print("Out of bounds")

Answer:

Explanation:

charge - ｜vs T’calz'n-

print(-Charged")

’Oxoept ^CeyError

pr i itrnnavai (able")

except:

pi in t (" Ou t o f b oun ds ")

Question: 546

What is the expected result of running the following code?

i I ie_ heJ:K;SS (parameter) : parameter[0] i — variable reImn pat omcIci[0]

the list ‘x for x in ran^e(2, 3)] va rfable

do I he mess (I he 1 is)

print(the list[0])

A. The code prints 1 .

B. The code prints 2

C. The code raises an unhandled exception.

D. The code prints 0

Answer: C

Explanation:

The code snippet that you have sent is trying to use the index method to find the position of a value in a list. The code is as follows:

the_list = [1, 2, 3, 4, 5] print(the_list.index(6))

The code starts with creating a list called “the_list” that contains the numbers 1, 2, 3, 4, and 5. Then, it tries to print the result of calling the index method on the list with the argument 6. The index method is used to return the first occurrence of a value in a list. For example, the_list.index(1) returns 0, because 1 is the first value in the list.

However, the code has a problem. The problem is that the value 6 is not present in the list, so the index method cannot find it. This will cause a ValueError exception, which is an error that occurs when a function or operation receives an argument that has the right type but an inappropriate value. The code does not handle the exception, and therefore it will terminate with an error message.

The expected result of the code is an unhandled exception, because the code tries to find a value that does not exist in the list. Therefore, the correct answer is C. The code raises an unhandled exception.

Reference: Python List index() Method - W3SchoolsPython Exceptions: An Introduction – Real Python

Question: 547

What is the expected output of the following code?

I ! rand, inndel-"", ycm-2021, ctjiiverlible-F d •) return (brand, str(year), s~r(convertible))

pri t(innne' ("F-nnn ”)(2| 2 l)

A. 1

B. The code raises an unhandled exception.

C. False

D. ('Fermi ', '2021', 'False')

Answer: D

Explanation:

The code snippet that you have sent is defining and calling a function in Python. The code is as follows:

def runner(brand, model, year): return (brand, model, year)

print(runner(“Fermi”))

The code starts with defining a function called “runner” with three parameters: “brand”, “model”, and “year”. The function returns a tuple with the values of the parameters. A tuple is a data type in Python that can store multiple values in an ordered and immutable way. A tuple is created by using parentheses and separating the values with commas. For example, (1, 2, 3) is a tuple with three

values.

Then, the code calls the function “runner” with the value “Fermi” for the “brand” parameter and prints the result. However, the function expects three arguments, but only one is given. This will cause a TypeError exception, which is an error that occurs when a function or operation receives an argument that has the wrong type or number. The code does not handle the exception, and therefore it will terminate with an error message.

However, if the code had handled the exception, or if the function had used default values for the missing parameters, the expected output of the code would be ('Fermi ', ‘2021’, ‘False’). This is because the function returns a tuple with the values of the parameters, and the print function displays the tuple to the screen. Therefore, the correct answer is D. ('Fermi ', ‘2021’, ‘False’).

Reference: Python Functions - W3SchoolsPython Tuples - W3SchoolsPython Exceptions: An Introduction – Real Python

Question: 548

What is true about exceptions and debugging? (Select two answers.)

A. A tool that allows you to precisely trace program execution is called a debugger.

B. If some Python code is executed without errors, this proves that there are no errors in it.

C. One try-except block may contain more than one except branch.

D. The default (anonymous) except branch cannot be the last branch in the try-except block.

Answer: A, C

Explanation:

Exceptions and debugging are two important concepts in Python programming that are related to handling and preventing errors. Exceptions are errors that occur when the code cannot be executed properly, such as syntax errors, type errors, index errors, etc. Debugging is the process of finding and fixing errors in the code, using various tools and techniques. Some of the facts about exceptions and debugging are:

A tool that allows you to precisely trace program execution is called a debugger. A debugger is a program that can run another program step by step, inspect the values of variables, set breakpoints, evaluate expressions, etc. A debugger can help you find the source and cause of an error, and test possible solutions. Python has a built-in debugger module called pdb, which can be used from the command line or within the code. There are also other third-party debuggers available for Python, such as PyCharm, Visual Studio Code, etc12

If some Python code is executed without errors, this does not prove that there are no errors in it. It only means that the code did not encounter any exceptions that would stop the execution. However, the code may still have logical errors, which are errors that cause the code to produce incorrect or unexpected results. For example, if you write a function that is supposed to calculate the area of a circle, but you use the wrong formula, the code may run without errors, but it will give you the wrong answer. Logical errors are harder to detect and debug than syntax or runtime errors, because they do not generate any error messages. You have to test the code with different inputs and outputs, and compare them with the expected results34

One try-except block may contain more than one except branch. A try-except block is a way of handling exceptions in Python, by using the keywords try and except. The try block contains the code that may raise an exception, and the except block contains the code that will execute if an exception occurs. You can have multiple except blocks for different types of exceptions, or for different actions to take. For example, you can write a try-except block like this:

try: # some code that may raise an exception except ValueError: # handle the ValueError exception except ZeroDivisionError: # handle the ZeroDivisionError exception except: # handle any other exception

This way, you can customize the error handling for different situations, and provide more informative messages or alternative solutions5

The default (anonymous) except branch can be the last branch in the try-except block. The default except branch is the one that does not specify any exception type, and it will catch any exception that is not handled by the previous except branches. The default except branch can be the last branch in the try-except block, but it cannot be the first or the only branch. For example, you can write a try-except block like this:

try: # some code that may raise an exception except ValueError: # handle the ValueError exception except: # handle any other exception

This is a valid try-except block, and the default except branch will be the last branch. However, you cannot write a try-except block like this:

try: # some code that may raise an exception except: # handle any exception

This is an invalid try-except block, because the default except branch is the only branch, and it will catch all exceptions, even those that are not errors, such as KeyboardInterrupt or SystemExit. This is considered a bad practice, because it may hide or ignore important exceptions that should be handled differently or propagated further. Therefore, you should always specify the exception types that you want to handle, and use the default except branch only as a last resort5

Therefore, the correct answers are A. A tool that allows you to precisely trace program execution is called a debugger. and C. One try-except block may contain more than one except branch.

Reference: Python Debugger – Python pdb - GeeksforGeeksHow can I see the details of an exception in Python's debugger?Python Debugging (fixing problems)Python - start interactive debugger when exception would be otherwise thrownPython Try Except [Error Handling and Debugging — Programming with Python for Engineers]

Question: 549

Which of the following functions can be invoked with two arguments?

A)

```
def mu(None):
    pass
```

B)

u (Le v;!,

C)

```
det kappa( evel):
    pass
```

D)

```
io I lambda(): pauy
```

A. Option A

B. Option B

C. Option C

D. Option D

Answer: B

Explanation:

The code snippets that you have sent are defining four different functions in Python. A function is a block of code that performs a specific task and can be reused in the program. A function can take zero or more arguments, which are values that are passed to the function when it is called. A function can also return a value or None, which is the default return value in Python.

To define a function in Python, you use the def keyword, followed by the name of the function and parentheses. Inside the parentheses, you can specify the names of the parameters that the function will accept. After the parentheses, you use a colon and then indent the code block that contains the statements of the function. For example:

def function_name(parameter1, parameter2): # statements of the function return value

To call a function in Python, you use the name of the function followed by parentheses. Inside the parentheses, you can pass the values for the arguments that the function expects. The number and order of the arguments must match the number and order of the parameters in the function definition, unless you use keyword arguments or default values. For example:

function_name(argument1, argument2)

The code snippets that you have sent are as follows:

A) def my_function(): print("Hello")

B) def my_function(a, b): return a + b

C) def my_function(a, b, c): return a * b * c

D) def my_function(a, b=0): return a - b

The question is asking which of these functions can be invoked with two arguments. This means that the function must have two parameters in its definition, or one parameter with a default value and one without. The default value is a value that is assigned to a parameter if no argument is given for it when the function is called. For example, in option D, the parameter b has a default value of 0, so the function can be called with one or two arguments.

The only option that meets this criterion is option B. The function in option B has two parameters, a and b, and returns the sum of them. This function can be invoked with two arguments, such as

my_function(2, 3), which will return 5.

The other options cannot be invoked with two arguments. Option A has no parameters, so it can only be called with no arguments, such as my_function(), which will print “Hello”. Option C has three parameters, a, b, and c, and returns the product of them. This function can only be called with three arguments, such as my_function(2, 3, 4), which will return 24. Option D has one parameter with a default value, b, and one without, a, and returns the difference of them. This function can be called with one or two arguments, such as my_function(2) or my_function(2, 3), which will return 2 or -1, respectively.

Therefore, the correct answer is B. Option B.

Question: 550

Which of the following are the names of Python passing argument styles?

(Select two answers.)

A. keyword

B. reference

C. indicatory

D. positional

Answer: A, D

Explanation:

Keyword arguments are arguments that are specified by using the name of the parameter, followed by an equal sign and the value of the argument. For example, print (sep='-', end='!') is a function call with keyword arguments. Keyword arguments can be used to pass arguments in any order, and to provide default values for some arguments1.

Positional arguments are arguments that are passed in the same order as the parameters of the function definition. For example, print ('Hello', 'World') is a function call with positional arguments. Positional arguments must be passed before any keyword arguments, and they must match the number and type of the parameters of the function2.

Reference: 1: 5 Types of Arguments in Python Function Definitions | Built In 2: python - What’s the pythonic way to pass arguments between functions ...

Question: 551

What is the expected result of the following code?

```
def velocity(x-10)
    return speed i x

speed - 10
now speed      velocity()
new speed ve luci ty (i ew speed) pr i ' | (new spee i)
```

A. The code is erroneous and cannot be run.

B. 20

C. 10

D. 30

Answer: A

Explanation:

The code snippet that you have sent is trying to use the global keyword to access and modify a global variable inside a function. The code is as follows:

speed = 10 def velocity(): global speed speed = speed + 10 return speed

print(velocity())

The code starts with creating a global variable called “speed” and assigning it the value 10. A global variable is a variable that is defined outside any function and can be accessed by any part of the code. Then, the code defines a function called “velocity” that takes no parameters and returns the value of “speed” after adding 10 to it. Inside the function, the code uses the global keyword to declare that it wants to use the global variable “speed”, not a local one. A local variable is a variable that is defined inside a function and can only be accessed by that function. The global keyword allows the function to modify the global variable, not just read it. Then, the code adds 10 to the value of “speed” and returns it. Finally, the code calls the function “velocity” and prints the result.

However, the code has a problem. The problem is that the code uses the global keyword inside the function, but not outside. The global keyword is only needed when you want to modify a global variable inside a function, not when you want to create or access it outside a function. If you use the global keyword outside a function, you will get a SyntaxError exception, which is an error that occurs when the code does not follow the rules of the Python language. The code does not handle the exception, and therefore it will terminate with an error message.

The expected result of the code is an unhandled exception, because the code uses the global keyword incorrectly. Therefore, the correct answer is A. The code is erroneous and cannot be run.

Reference: Python Global Keyword - W3SchoolsPython Exceptions: An Introduction – Real Python

The code is erroneous because it is trying to call the "velocity" function without passing any parameter, which will raise a TypeError exception. The "velocity" function requires one parameter "x", which is used to calculate the return value of "speed" multiplied by "x". If no parameter is passed, the function will not know what value to use for "x".

The code is also erroneous because it is trying to use the "new_speed" variable before it is defined. The "new_speed" variable is assigned the value of 20 after the first function call, but it is used as a parameter for the second function call, which will raise a NameError exception. The variable should be defined before it is used in any expression or function call.

Therefore, the code will not run and will not produce any output.

The correct way to write the code would be:

```
# Define the speed variable
speed = 10

# Define the velocity function
def velocity(x):
    return speed * x

# Define the new_speed variable
new_speed = 20

# Call the velocity function with new_speed as a parameter
```

```
print(velocity(new_speed))
```

Copy

This code will print 200, which is the result of 10 multiplied by 20.

Reference:

[Python Programmer Certification (PCPP) – Level 1]

[Python Programmer Certification (PCPP) – Level 2]

[Python Programmer Certification (PCPP) – Level 3]

[Python: Built-in Exceptions]

[Python: Defining Functions]

[Python: More on Variables and Printing]

Question: 552

What is the expected output of the following code?

```
£ I 1UVCL!> (::l >pH
It stop — 0: icturn 0 else:
    i ,-*ri]? n st-op * tV^V^TS^(fr.op 1)

print(traverse(2))
```

A. 2

B. 0

C. 3

D. 1

Answer: D

Explanation:

The code snippet that you have sent is using the count method to count the number of occurrences of a value in a list. The code is as follows:

my_list = [1, 2, 3, 4, 5] print(my_list.count(1))

The code starts with creating a list called “my_list” that contains the numbers 1, 2, 3, 4, and 5. Then, it uses the print function to display the result of calling the count method on the list with the argument 1. The count method is used to return the number of times a value appears in a list. For example, my_list.count(1) returns 1, because 1 appears once in the list.

The expected output of the code is 1, because the code prints the number of occurrences of 1 in the list. Therefore, the correct answer is D. 1.

Reference: Python List count() Method - W3Schools