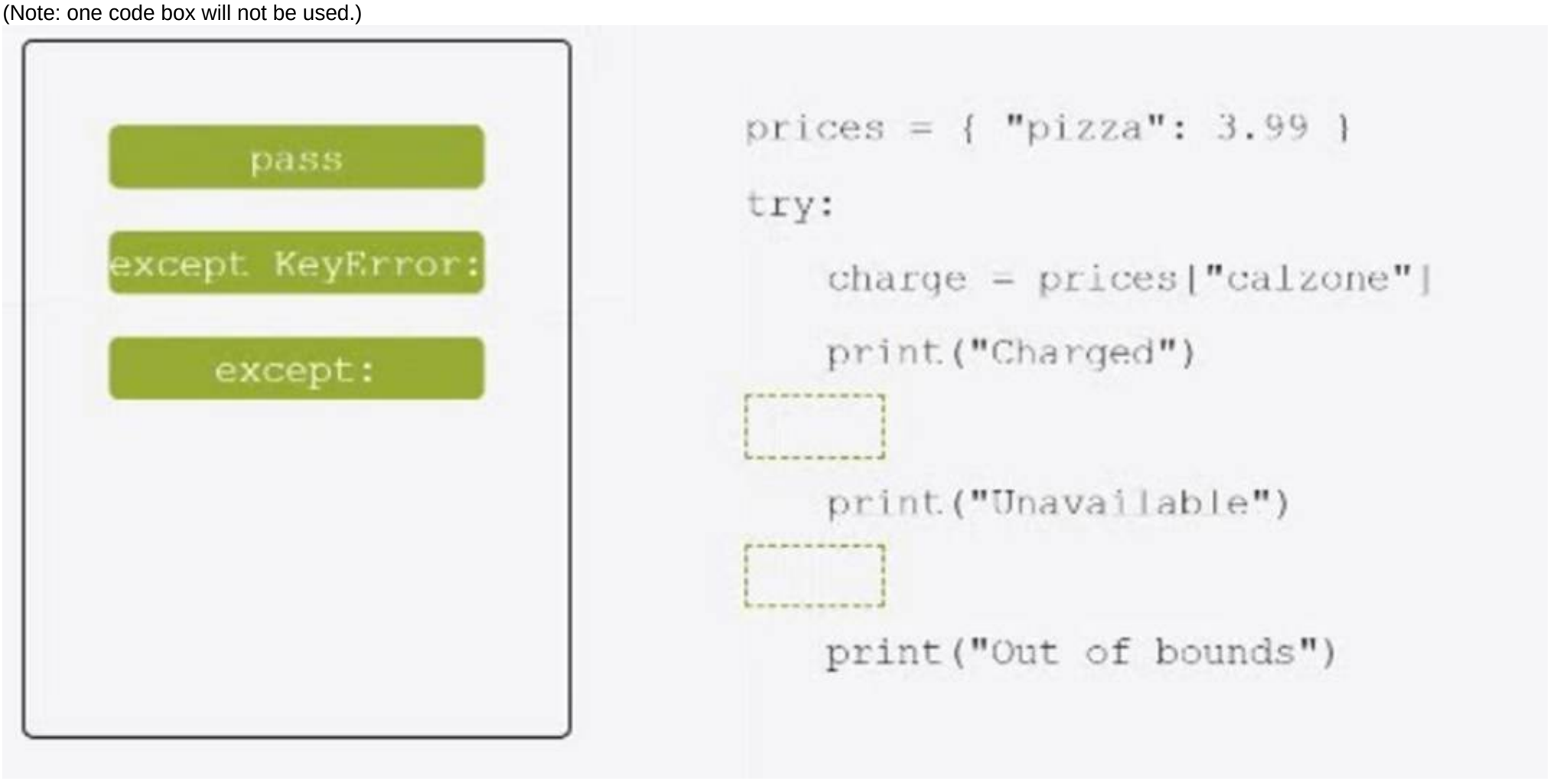
Exam Questions PCEP-30-02

PCEP - Certified Entry-Level Python Programmer

https://www.2passeasy.com/dumps/PCEP-30-02/

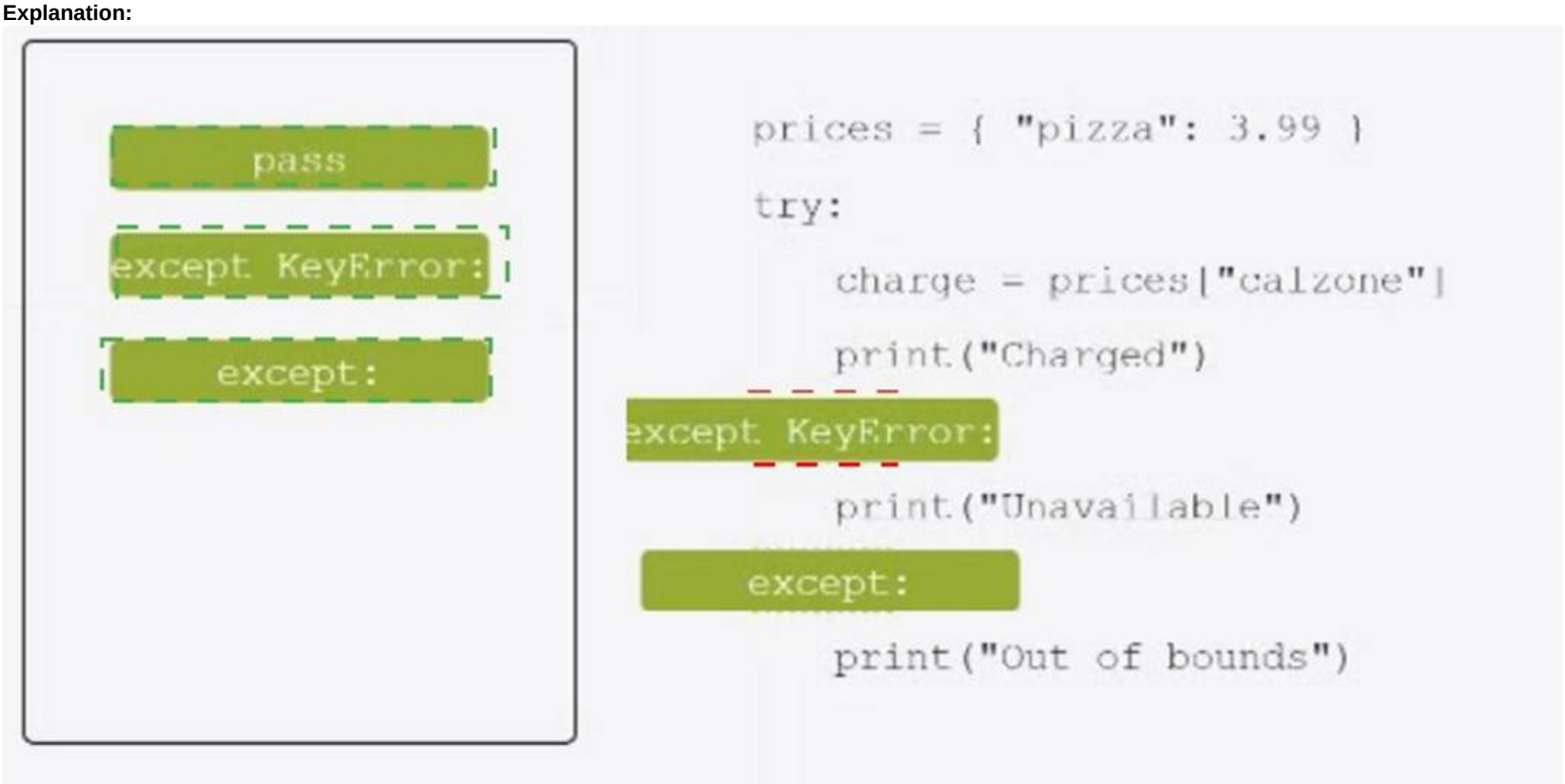
NEW QUESTION 1
DRAG DROP
Drag and drop the code boxes in order to build a program which prints Unavailable to the screen.
(Note: one code box will not be used.)



A. Mastered
B. Not Mastered

Answer: A

Explanation:



NEW QUESTION 2
What happens when the user runs the following code?

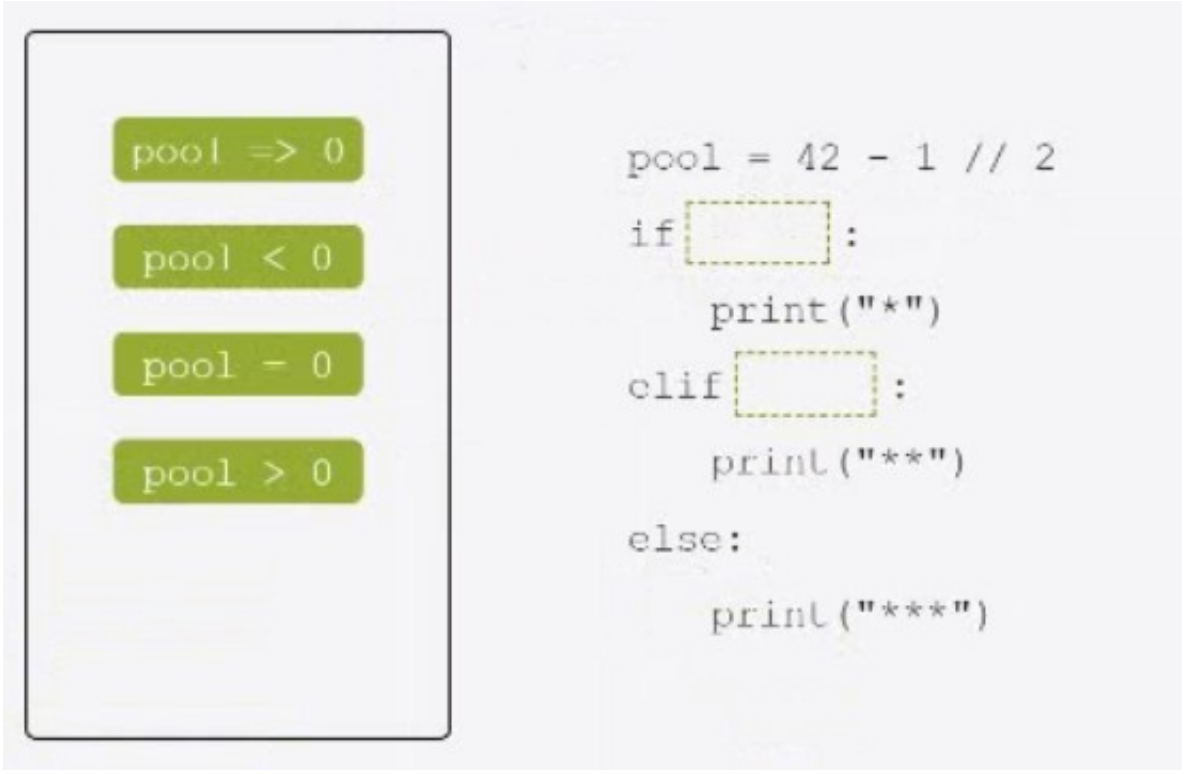
```
lolal = 0
for i in range(4):
    if 2 * i < 4:
        total += 1
else:
    total += 1
print(total)
```

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]

NEW QUESTION 3
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.)



A. Mastered
B. Not Mastered

Answer: A

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]

NEW QUESTION 4
What is the expected result of the following code?

```
def velocity(x=10):
    return speed | x

speed = 10
new_speed = velocity()
new_speed = velocity(new_speed)
print(new_speed)
```

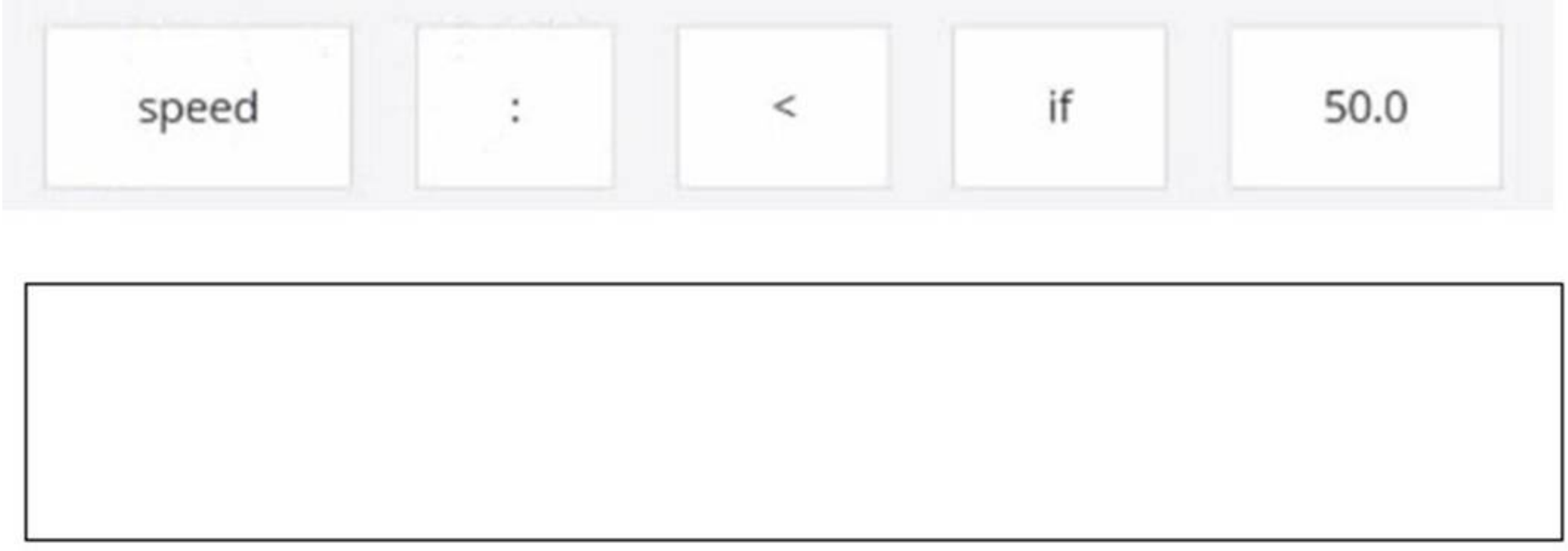
A. The code is erroneous and cannot be run.

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. References:
[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]

NEW QUESTION 5
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.



A. Mastered
B. Not Mastered

Answer: A

Explanation:
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

NEW QUESTION 6
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]

NEW QUESTION 7
What is the expected output of the following code?

```
def runner(brand, model="", year=2021, convertible=False):
    return (brand, str(year), str(convertible))


print(runner("Fermi")[2][2])
```

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

NEW QUESTION 8
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: BD

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.
* D. 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]

NEW QUESTION 9
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: AD

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

NEW QUESTION 10
.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual PCEP-30-02 Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the PCEP-30-02 Product From:

https://www.2passeasy.com/dumps/PCEP-30-02/

Money Back Guarantee

PCEP-30-02 Practice Exam Features:

* PCEP-30-02 Questions and Answers Updated Frequently

* PCEP-30-02 Practice Questions Verified by Expert Senior Certified Staff

* PCEP-30-02 Most Realistic Questions that Guarantee you a Pass on Your FirstTry

* PCEP-30-02 Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year