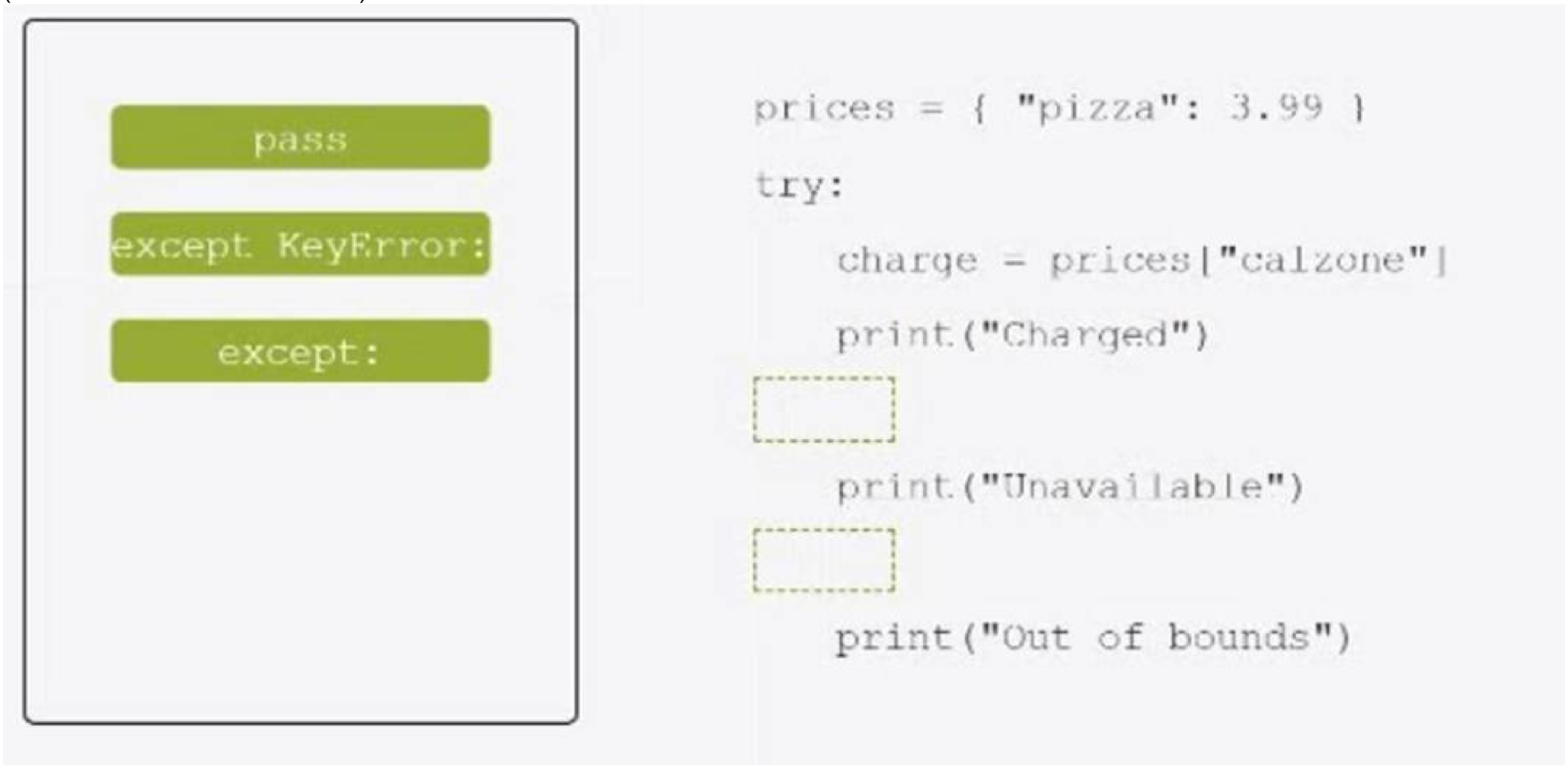
Exam Questions PCEP-30-02

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.)

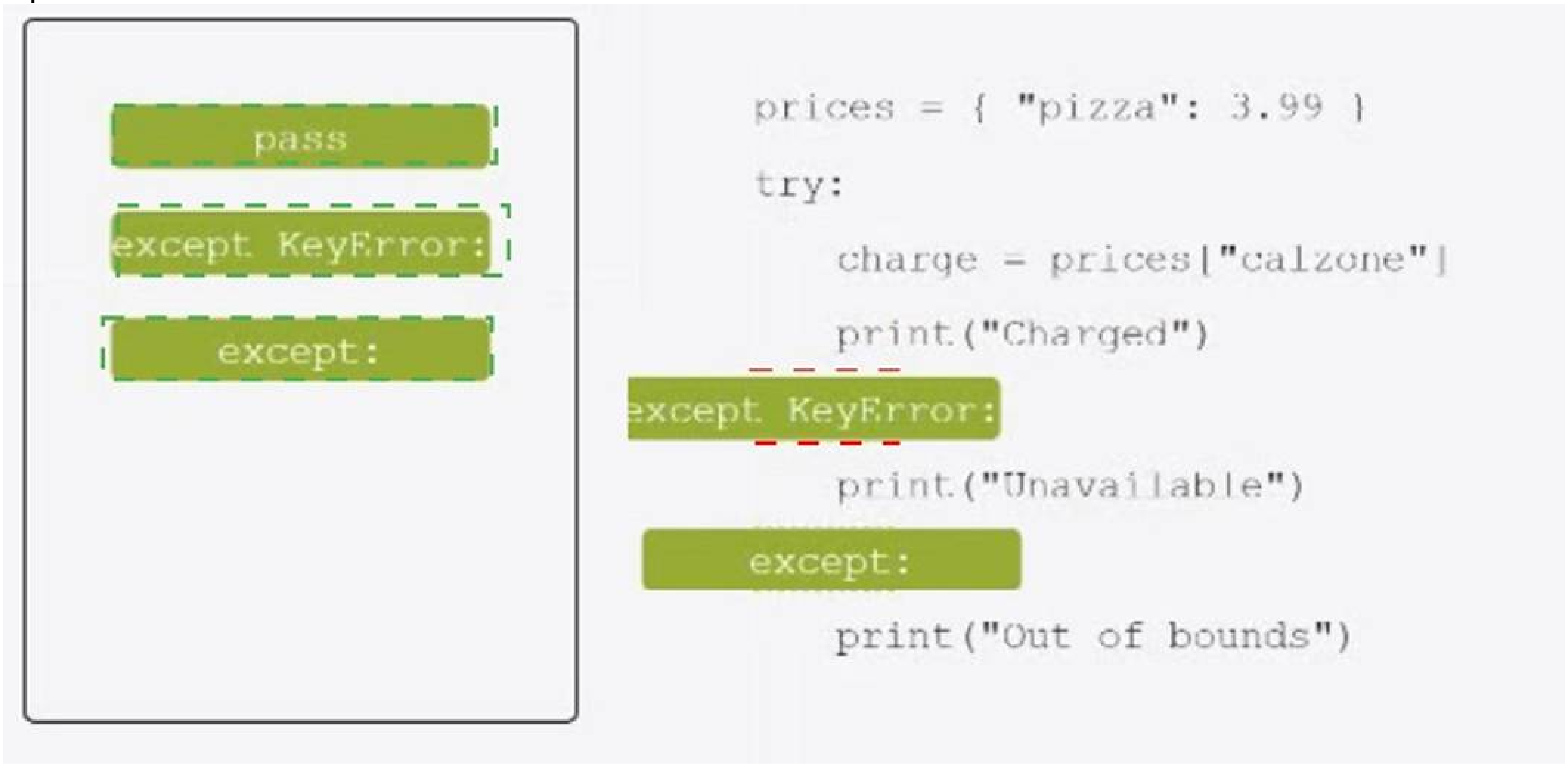


```
prices = { "pizza": 3.99 }
try:
    charge = prices["calzone"]
    print("Charged")

    print("Unavailable")

    print("Out of bounds")
```

A. Mastered
B. Not Mastered

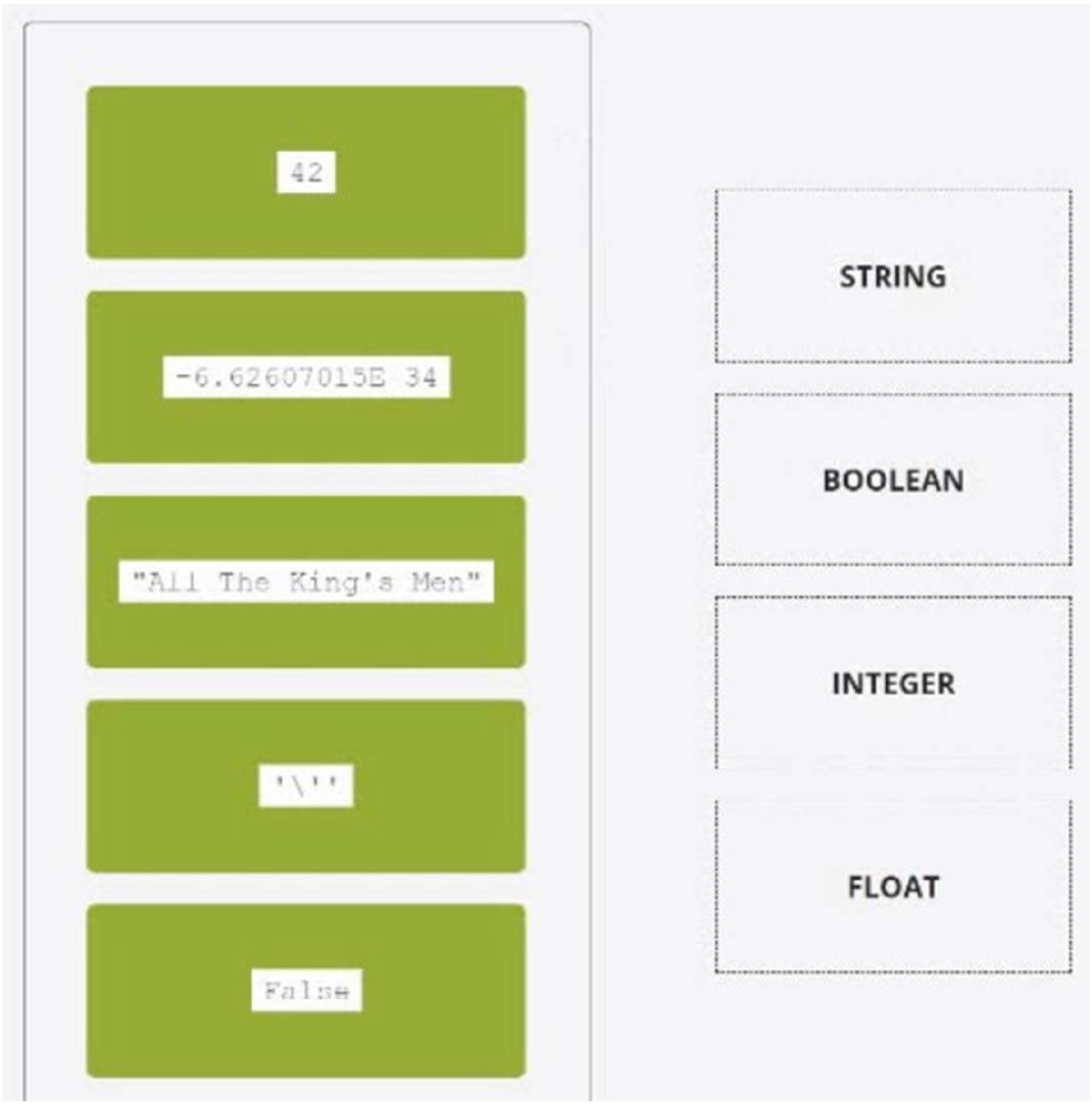
Answer: A

Explanation:



```
prices = { "pizza": 3.99 }
try:
    charge = prices["calzone"]
    print("Charged")
except KeyError:
    print("Unavailable")
except:
    print("Out of bounds")
```

NEW QUESTION 2
DRAG DROP
Drag and drop the literals to match their data type names.


42

-6.62607015E 34

"All The King's Men"

'\''

False

A. Mastered
B. Not Mastered

Answer: A

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]

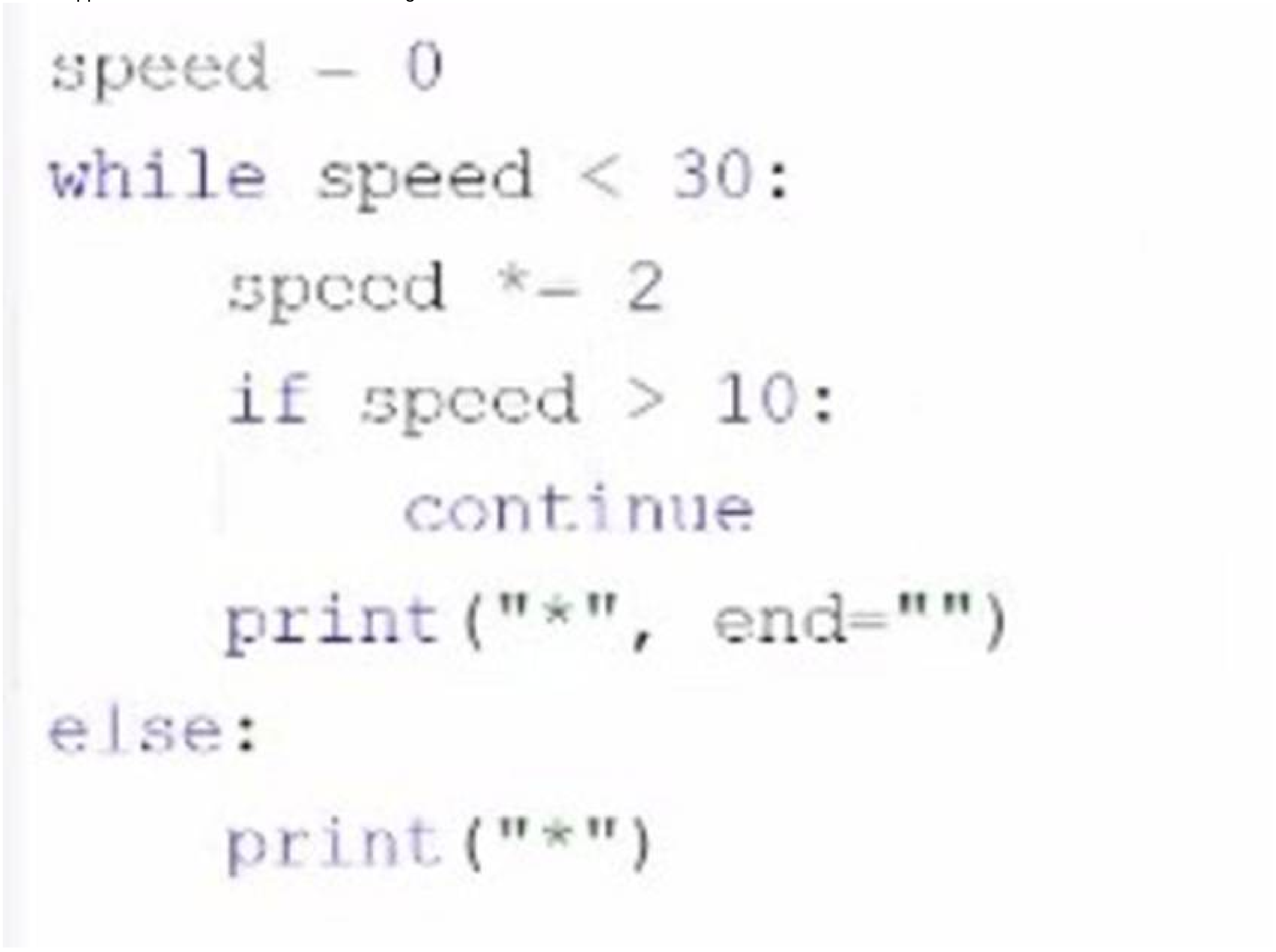
NEW QUESTION 3
A set of rules which defines the ways in which words can be coupled in sentences is called:

A. lexis
B. syntax
C. semantics
D. dictionary

Answer: B

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]

NEW QUESTION 4
What happens when the user runs the following code?



```
speed - 0
while speed < 30:
    speed *= 2
    if speed > 10:
        continue
    print("*", end="")
else:
    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 1. 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]

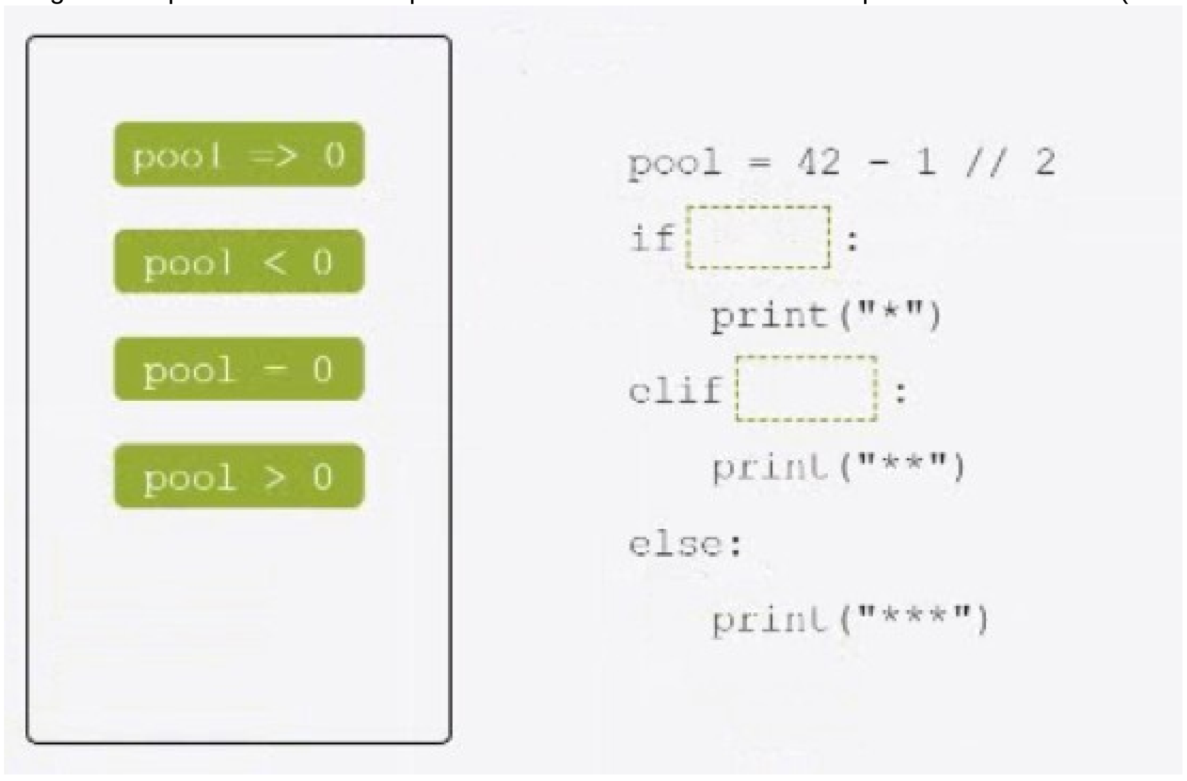
NEW QUESTION 5
Which of the following are the names of Python passing argument styles? (Select two answers.)

A. keyword
B. reference
C. indicatory
D. positional

Answer: AD

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.
References: 1: 5 Types of Arguments in Python Function Definitions | Built In 2: python - What??s the pythonic way to pass arguments between functions ??

NEW QUESTION 6
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.)



```
pool = 42 - 1 // 2
if [    ]:
    print("*")
elif [    ]:
    print("**")
else:
    print("***")
```

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 7
What is true about exceptions and debugging? (Select two answers.)

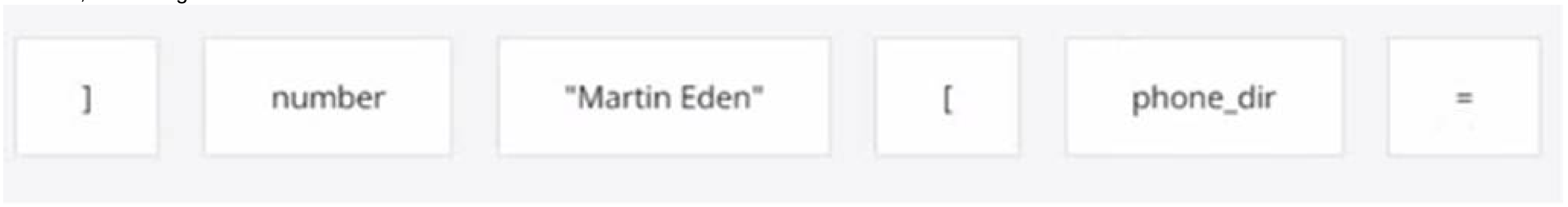
A. A tool that allows you to precisely trace program execution is called a debugger.
B. If some Python code is executed without errors, this proves that there are no errors in it.
C. One try-except block may contain more than one except branch.
D. The default (anonymous) except branch cannot be the last branch in the try-except block.

Answer: AC

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]

NEW QUESTION 8
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.



A. Mastered
B. Not Mastered

Answer: A

Explanation:
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 ??]

NEW QUESTION 9
Python Is an example of which programming language category?

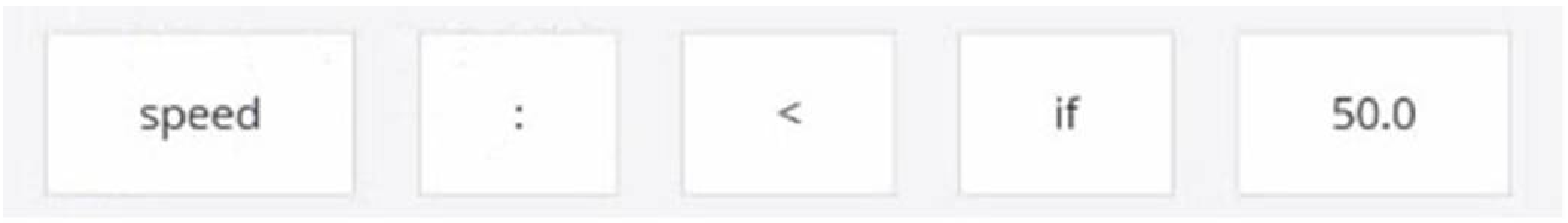
A. interpreted
B. assembly
C. compiled
D. machine

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]

NEW QUESTION 10
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.

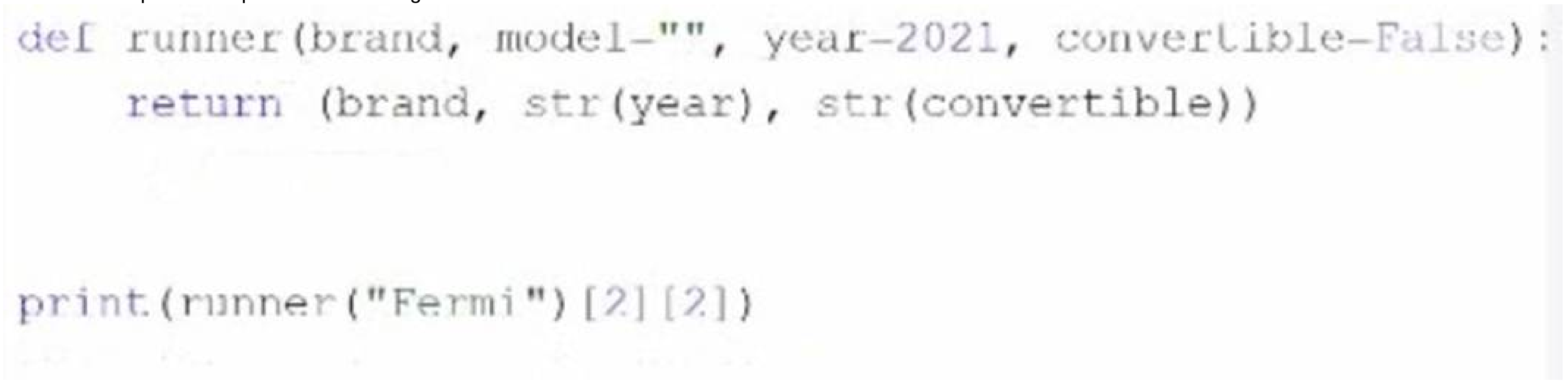


A. Mastered
B. Not Mastered

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 10
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])
```

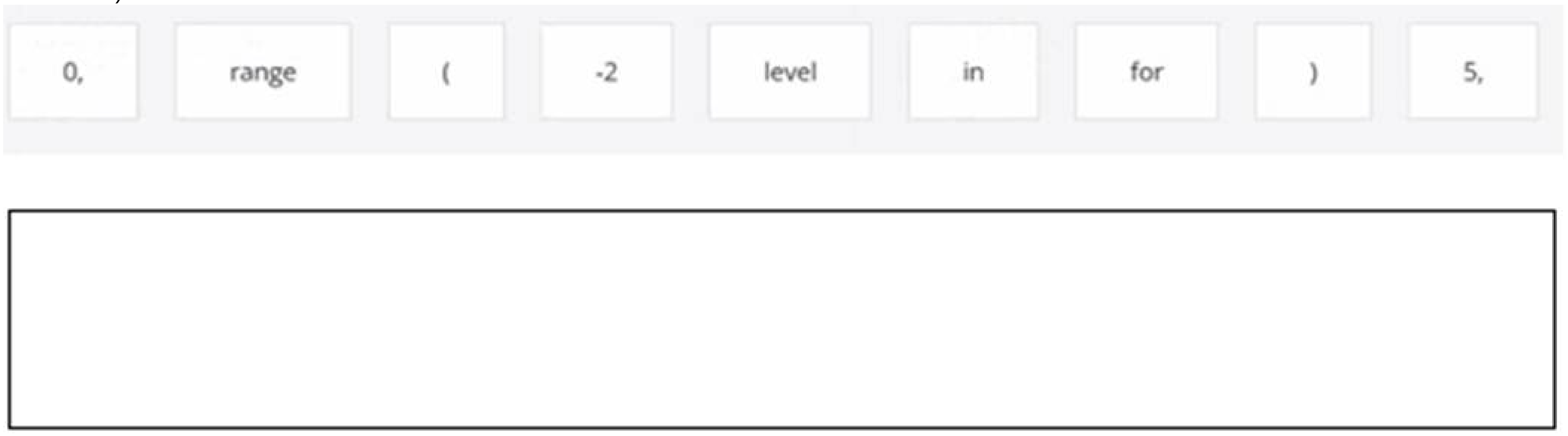
A. 1
B. The code raises an unhandled exception.
C. False
D. ('Fermi ', '2021', 'False')

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

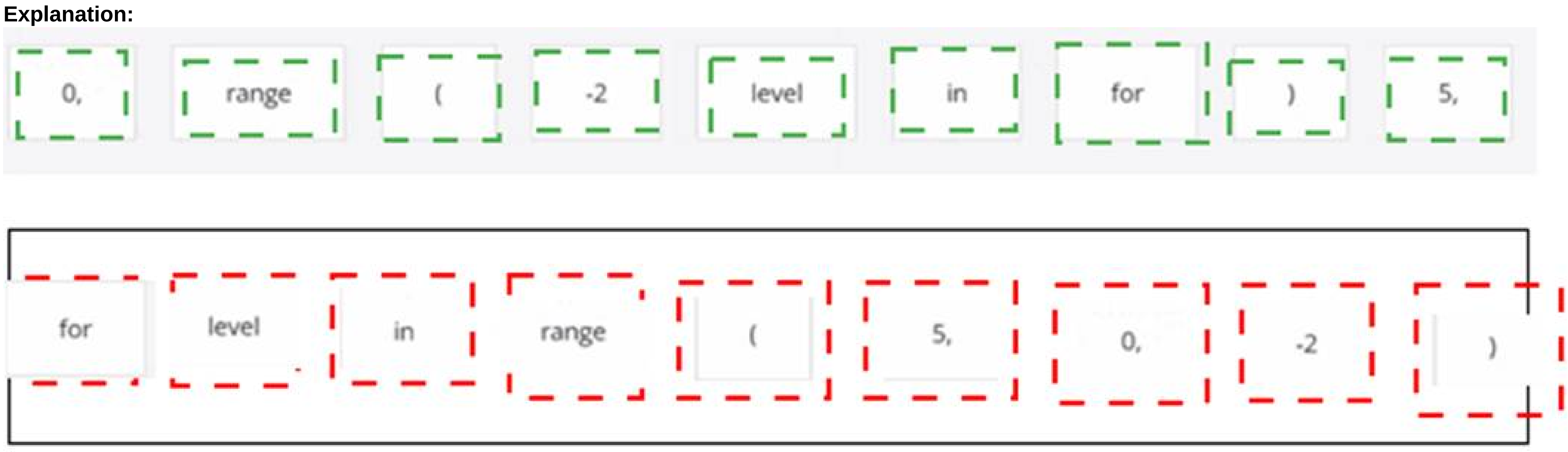
NEW QUESTION 12
DRAG DROP
Arrange the code boxes in the correct positions in order to obtain a loop which executes its body with the level variable going through values 5, 1, and 1 (in the same order).



A. Mastered
B. Not Mastered

Answer: A

Explanation:



NEW QUESTION 17
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 22
......