Python-Institute

Exam Questions PCEP-30-02

PCEP - Certified Entry-Level Python Programmer

NEW QUESTION 1
What is the expected output of the following code?

```
collection = []
collection.append(1)
collection.insert(0, 2)
duplicate = collection
duplicate.append(3)
print(len(collection) + len(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]

NEW QUESTION 2
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 3
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 4
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]

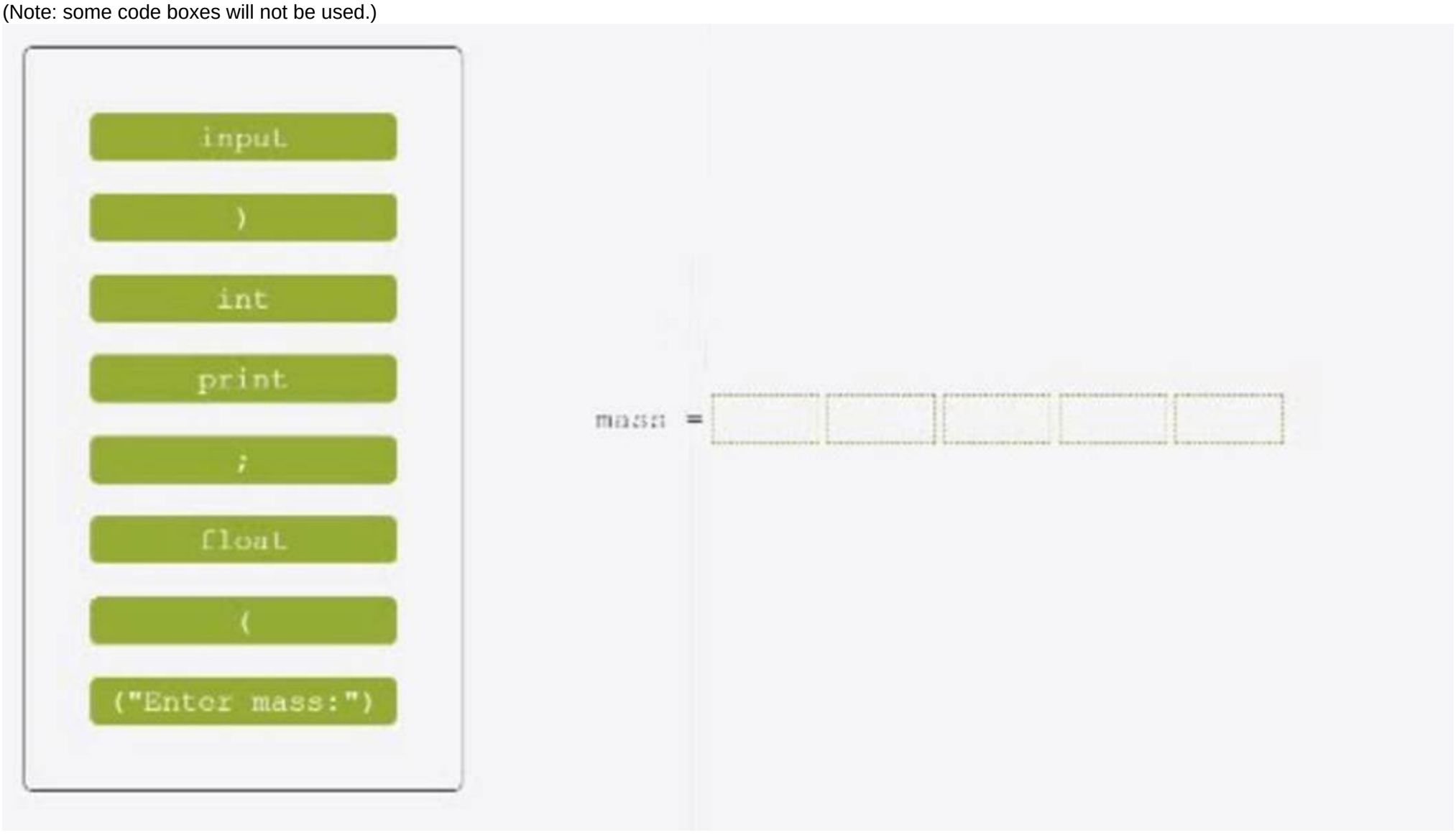
NEW QUESTION 5
Python Is an example of which programming language category?

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]

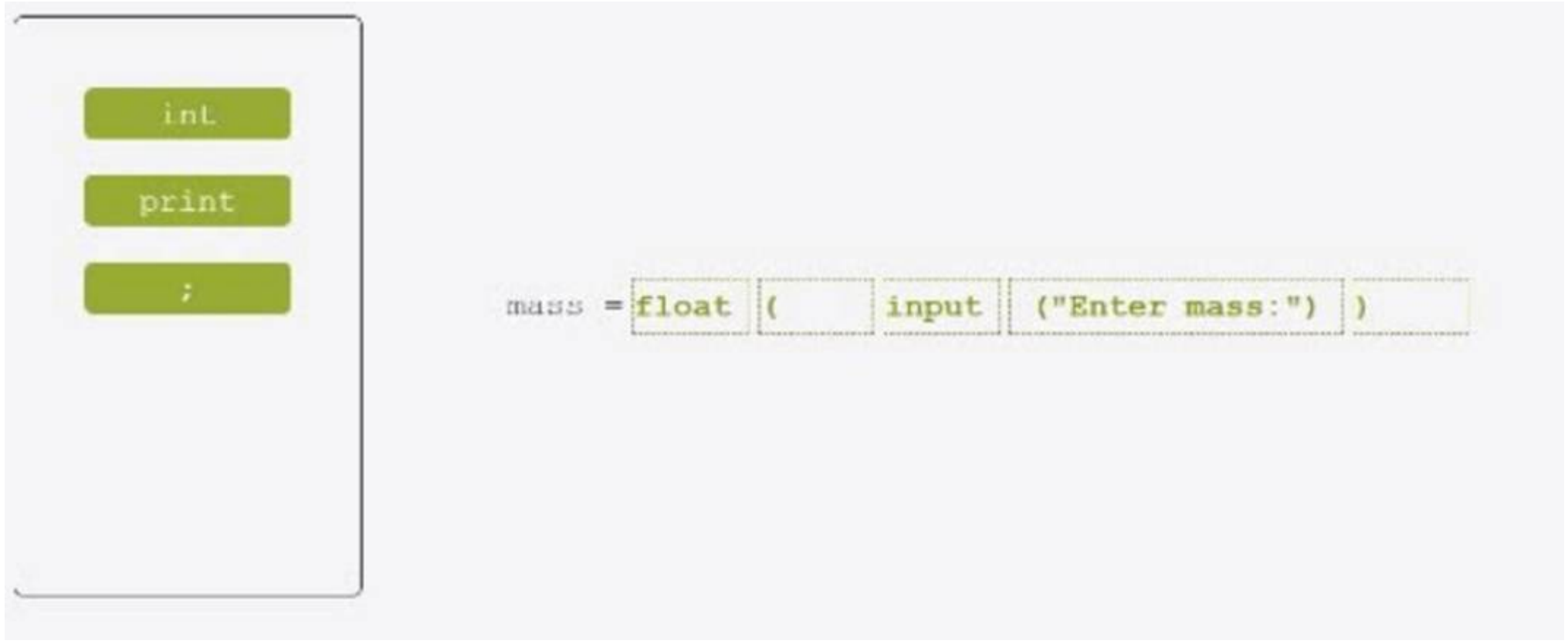
NEW QUESTION 6
DRAG DROP
Insert the code boxes in the correct positions in order to build a line of code which asks the user for a float value and assigns it to the mass variable.
(Note: some code boxes will not be used.)



A. Mastered
B. Not Mastered

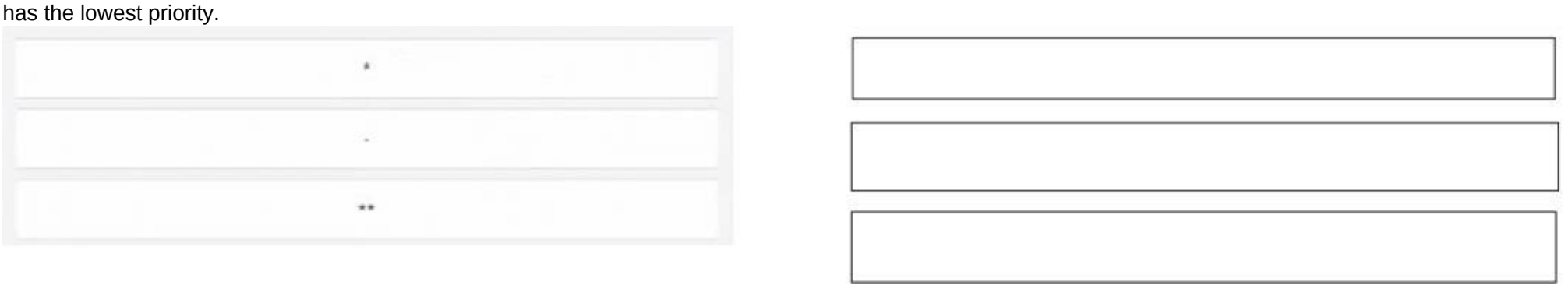
Answer: A

Explanation:



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]

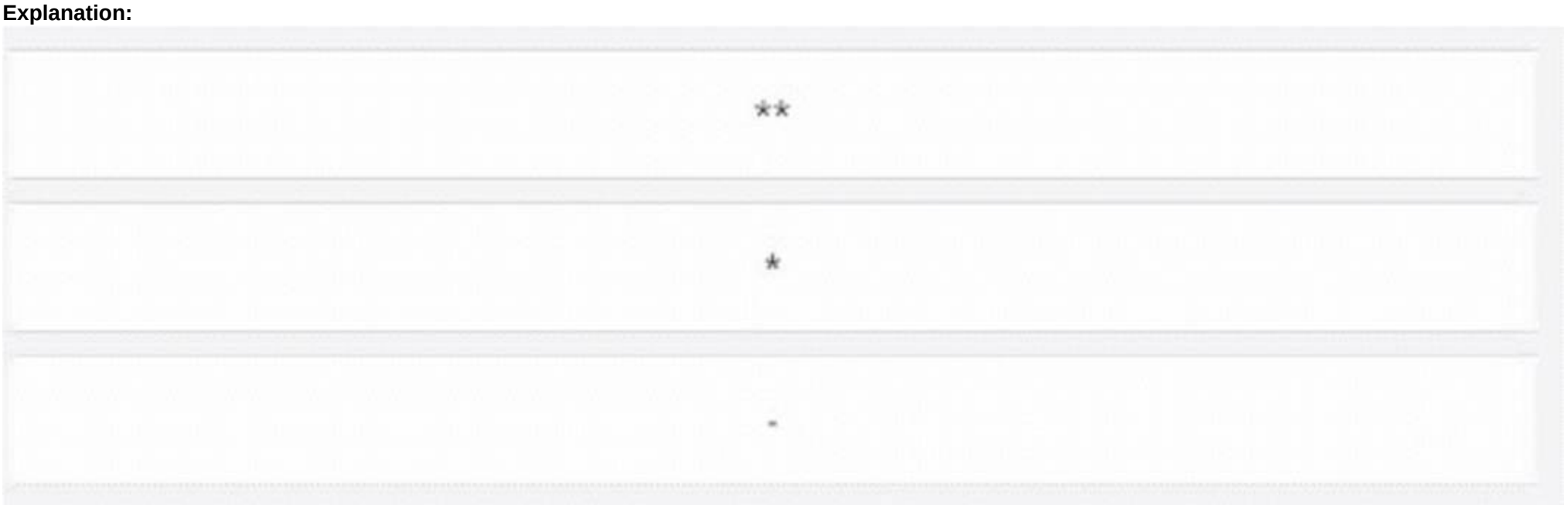
NEW QUESTION 7
DRAG DROP
Arrange the binary numeric operators in the order which reflects their priorities, where the top-most position has the highest priority and the bottom-most position has the lowest priority.



A. Mastered
B. Not Mastered

Answer: A

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

NEW QUESTION 8
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
.....

Thank You for Trying Our Product

We offer two products:

1st - We have Practice Tests Software with Actual Exam Questions

2nd - Questons and Answers in PDF Format

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