

Python-Institute

Exam Questions PCEP-30-02

PCEP - Certified Entry-Level Python Programmer



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.)

pass

except: KeyError:

except:

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

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

pass

except: KeyError:

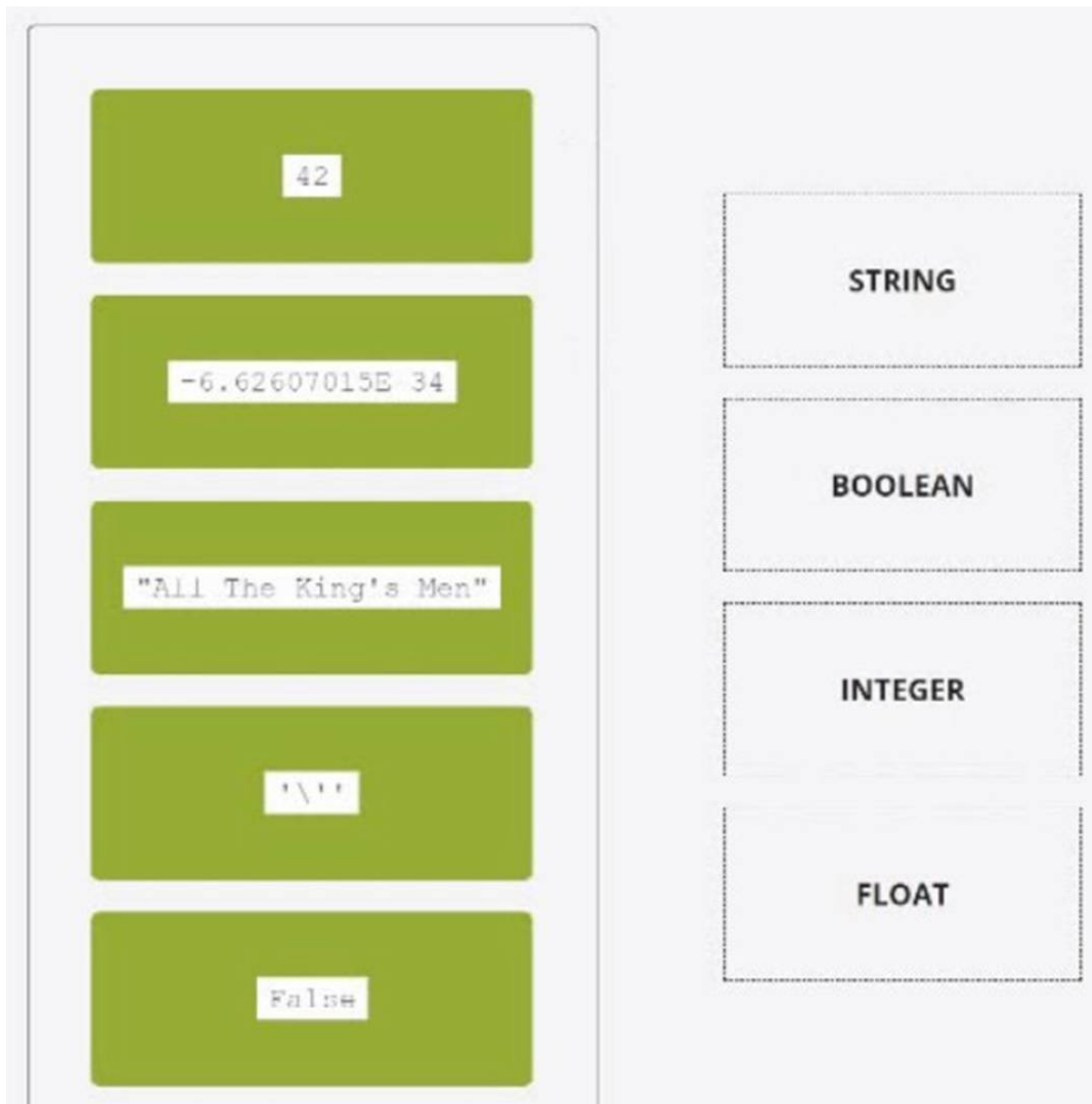
except:

```
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.



- 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

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.

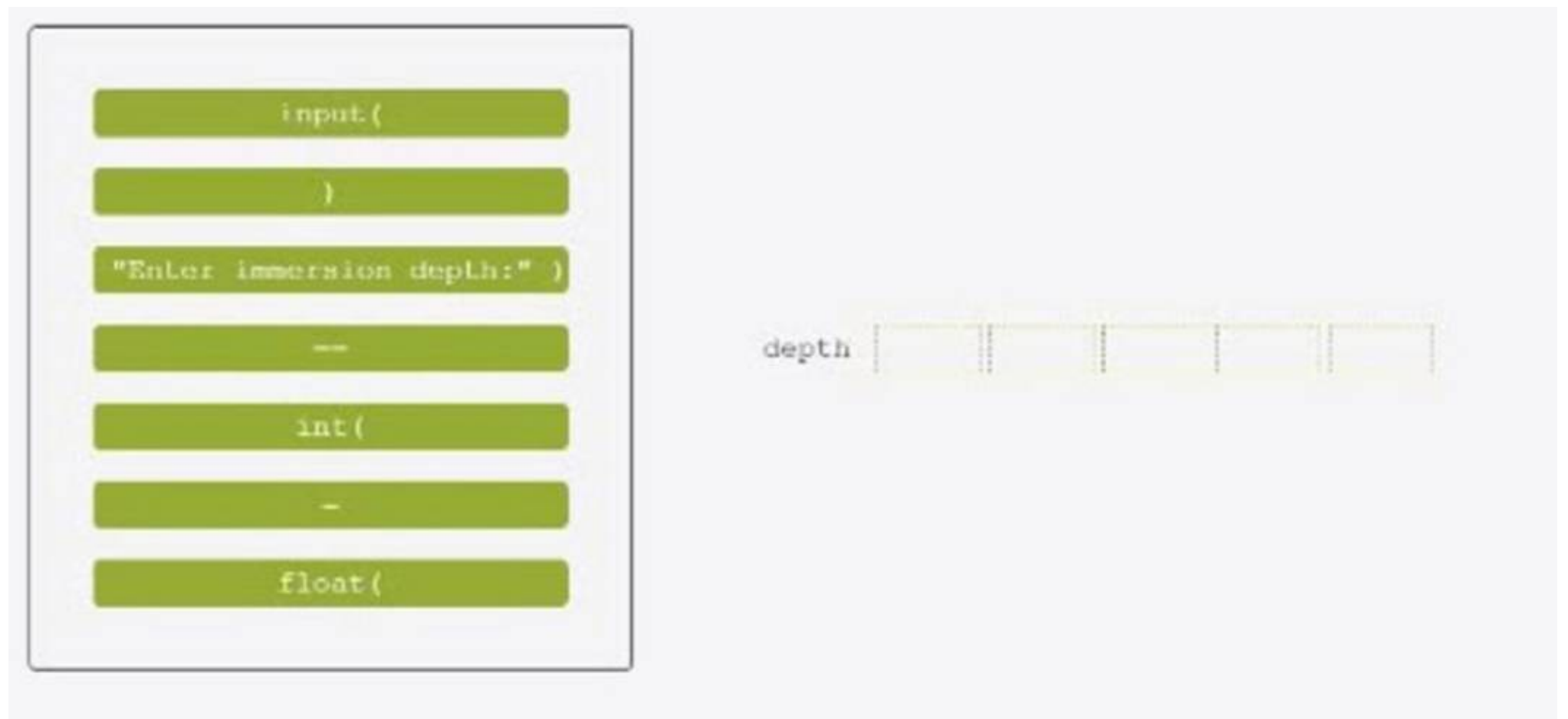
Reference: [Python Institute - Entry-Level Python Programmer Certification]

NEW QUESTION 4

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.)



- 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 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]

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

What is the expected output of the following code?

```
def traverse(stop):
    if stop == 0:
        return 0
    else:
        return stop * traverse(stop - 1)

print(traverse(2))
```

- A. 2
- B. 3
- C. 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

NEW QUESTION 7

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 == 0

pool < 0

pool = 0

pool > 0

```
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 8

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 9

DRAG DROP

Assuming that the phone_dir dictionary contains namenumber 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"

[

phone_dir

=

- 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 10

How many hashes (+) does the code output to the screen?

```
floor = 10
while floor != 0:
    floor //= 4
    print("#", end="")
else:
    print("#")
```

- 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 = 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 `floor` 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 `floor` 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]

NEW QUESTION 10

Assuming that the following assignment has been successfully executed:

```
the_list = ["1", 1, 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_lis`
- E. `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

NEW QUESTION 13

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 15

.....

Thank You for Trying Our Product

We offer two products:

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

2nd - Questions 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 First Try
- * PCEP-30-02 Practice Test Questions in Multiple Choice Formats and Updates for 1 Year

100% Actual & Verified — Instant Download, Please Click
[Order The PCEP-30-02 Practice Test Here](#)