Python-Institute

Exam Questions PCEP-30-02

PCEP - Certified Entry-Level Python Programmer

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

```
rates = (1.2, 1.4, 1.0)
new = rates[3:]
for rate in rates[-2:]:
    new += (rate,)
print(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

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
What is the expected output of the following code?

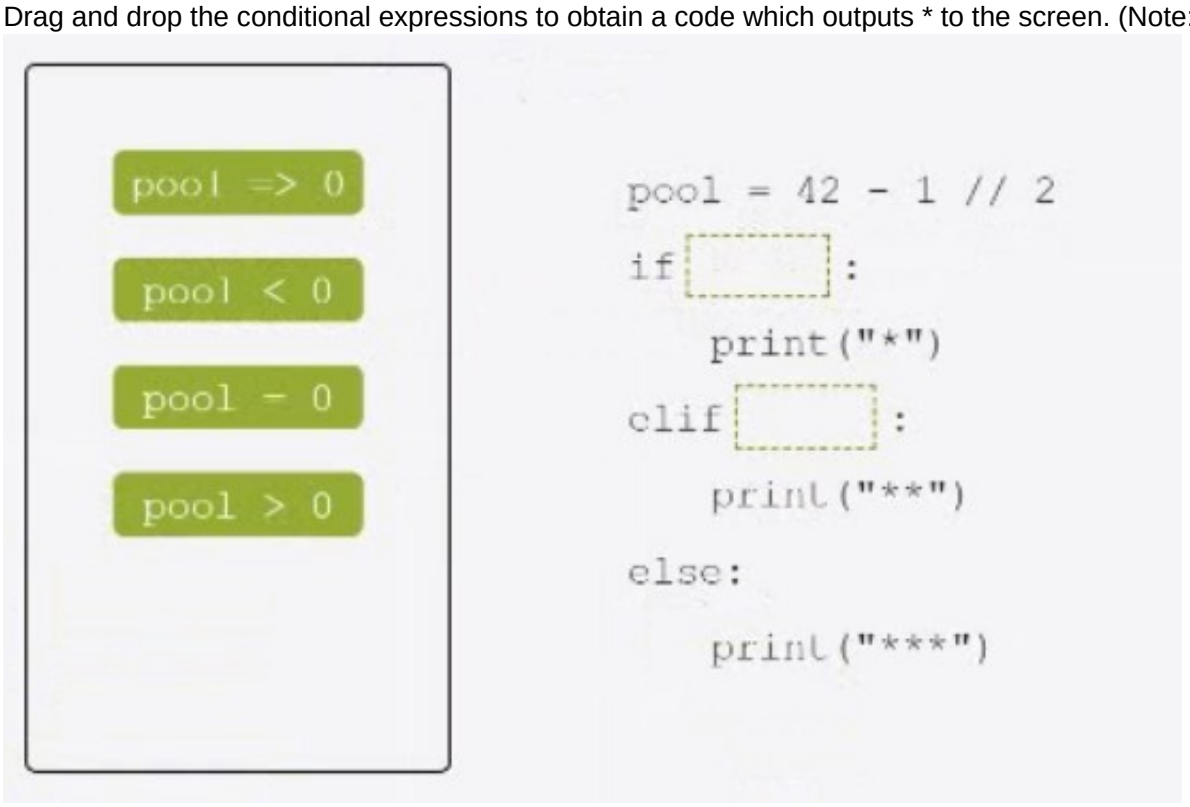
```
collection = []
collection.append(1)
collection.insert(0, 2)
duplicate = collection
duplicate.append(3)
print(len(collection) + len(duplicate))
```

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
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 5
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]

NEW QUESTION 6
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 7

Which of the following functions can be invoked with two arguments?
A)

```
def mu(Nonc):
    pass
```

B)

```
def iota(level, size = 0):
    pass
```

C)

```
def kappa(level):
    pass
```

D)

```
def lambda():
    pass
```

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.

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