Up-to-date Practice Test with Latest Questions and Answers covering latest syllabus and topics of the exam. Makes you ready to face actual exam.

KILL EXAMS

PCEP-30-02 Practice Questions
PCEP-30-02 Practice Test
PCEP-30-02 Practice Exam
PCEP-30-02 Exam Questions
PCEP-30-02 Study Guide

PASS

killexams.com

Python-Institute

PCEP-30-02

PCEP - Certified Entry-Level Python Programmer (PCEP)

ORDER FULL VERSION

https://killexams.com/pass4sure/exam-detail/PCEP-30-02

Question: 427

What will be the output of the following code snippet: print(type([]) is list)?

A. True
B. False
C. None
D. TypeError

Answer: A

Explanation:
The expression type([]) returns the type of an empty list, which is list. The comparison is checks if both operands refer to the same object, and since they do, the output will be True.

Question: 428

Which of the following Python statements will correctly create a list containing the numbers 1 to 10 inclusive?

A. numbers = [1:10]
B. numbers = list(range(1, 11))
C. numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
D. numbers = (1, 10)

Answer: B

Explanation:
To create a list of numbers from 1 to 10 inclusive, we can use the range function with list() to convert it into a list.

numbers = list(range(1, 11)) produces [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
The other options either use incorrect syntax or create a tuple.

Question: 429

What will be the output of the following code that uses the filter() function?
nums = [1, 2, 3, 4, 5]
even_nums = list(filter(lambda x: x % 2 == 0, nums))

print(even_nums)

A. [1, 2, 3]
B. [2, 4]
C. [1, 3, 5]
D. [1, 2, 3, 4, 5]

Answer: B

Explanation: The filter() function applies the lambda function to each element in nums, keeping only even numbers. The resulting list is [2, 4].

Question: 430

What will be the output of the following code?

```
def f(a, b=1, c=2):
return a * b + c

print(f(5, 2))
```

A. 12
B. 15
C. 20
D. 8
E. 9
F. The function will raise an error.

Answer: A

Explanation:
In this function, f takes three parameters, with b and c having default values. When called with 5 for a and 2 for b, it calculates 5 * 2 + 2, which equals 12.

Question: 431

What will be the output of the following code when executed?

```
def my_function(x):
return x + 1
```

```
result = my_function(3)
print(result)
```

A. 2
B. 3
C. 4
D. None

Answer: C

Explanation:

The function my_function takes an argument x, adds 1, and returns the result. For the input 3, it returns 4.

Question: 432

Consider the following snippet of code. What will be printed when the function func is called with the argument 4?

```
def func(n):
return (n + 1) if n % 2 == 0 else (n - 1)

print(func(4))
```

A. 3
B. 4
C. 5
D. 2
E. None

Answer: C

Explanation:
When func(4) is called, since 4 is even (4 % 2 == 0), the function returns 4 + 1, which equals 5.
Therefore, the printed result is 5.

Question: 433

What will be printed by the following code block?

```
def func(a, b):
a += b
return a
x = 10
y = 5
print(func(x, y))
print(x)
```

A. 15, 10
B. 15, 5
C. 10, 5
D. 15, 15

Answer: A

Explanation:
The function func adds b to a and returns the result. It does not modify the original x, so x remains 10.

Question: 434

What is the output of this code?

```
def square_numbers(lst):
return [x**2 for x in lst if x > 0]
print(square_numbers([-1, 0, 1, 2, 3]))
```

A. [1, 4, 9]
B. [0, 1, 4, 9]
C. [1, 2, 3]
D. [2, 3]
E. [-1, 0]

Answer: A

Explanation:

The square_numbers function squares only the positive numbers in the input list. For [-1, 0, 1, 2, 3], the positive numbers are 1, 2, and 3, leading to the output [1, 4, 9].

Question: 435

What does the following code output?

```
def check_value(x):
return "Positive" if x > 0 else "Negative" if x < 0 else "Zero"
print(check_value(-10))
print(check_value(0))
print(check_value(10))
```

A. Positive, Negative, Zero
B. Negative, Zero, Positive
C. Zero, Negative, Positive
D. Negative, Positive, Zero

Answer: B

Explanation:
The function checks the value of x and returns "Negative", "Zero", or "Positive" depending on its condition. The outputs correspond correctly to the inputs.

Question: 436

Which of the following statements best describes the purpose of the __init__ method in a Python class?

A. It is used to return a string representation of the object.
B. It initializes the object's attributes when an instance is created.
C. It is invoked when a class is inherited.
D. It is used to delete an instance of the object.

Answer: B

Explanation:
The __init__ method is a special method in Python classes that is automatically called when a new instance of the class is created. It allows you to set the initial state of the object by assigning values to its attributes.

Question: 437

A financial institution is developing a Python program to calculate the compound interest on an investment. The program should account for user inputs and provide the final amount after the specified time. You need to complete the code to meet the requirements.
principal = XXX

```
rate = YYY
time = ZZZ
amount = principal * (1 + rate / 100) ** time
print('The final amount after {} years is: {:.2f}'.format(time, amount))
```
What should you insert instead of XXX, YYY, and ZZZ?

A. XXX -> float(input('Enter the principal amount: ')) YYY -> float(input('Enter the interest rate: ')) ZZZ -> int(input('Enter the time in years: '))
B. XXX -> input('Enter the principal amount: ') YYY -> float(input('Enter the interest rate: ')) ZZZ -> int(input('Enter the time in years: '))
C. XXX -> float(input('Enter the principal amount: ')) YYY -> input('Enter the interest rate: ') ZZZ -> int(input('Enter the time in years: '))
D. XXX -> float(input('Enter the principal amount: ')) YYY -> float(input('Enter the interest rate: ')) ZZZ -> input('Enter the time in years: ')

Answer: A

Explanation:
```
principal = float(input('Enter the principal amount: '))
rate = float(input('Enter the interest rate: '))
time = int(input('Enter the time in years: '))
amount = principal * (1 + rate / 100) ** time
print('The final amount after {} years is: {:.2f}'.format(time, amount))
```
The program captures user inputs for principal and rate as floats, and time as an integer.
It calculates the compound interest and formats the output to two decimal places.

Question: 438

Which of the following statements about default parameters in Python functions is true?

A. Default parameters must be specified in every function call.
B. You cannot use mutable types as default parameters.
C. Default parameters are evaluated only once at function definition time.

Answer: C

Explanation:
Default parameters in Python are evaluated once when the function is defined, not each time the function is called.

Question: 439

What will be the output of the following code snippet involving a class method and class variable?
class Counter:
count = 0
def increment(self):
Counter.count += 1
counter1 = Counter()
counter1.increment()
counter2 = Counter()
counter2.increment()
print(Counter.count)

A. 1
B. 2
C. 0
D. Error

Answer: B

Explanation:
The class variable count is shared among all instances of the Counter class. Each time increment() is called, count increases by 1. After two calls, the count is 2.

Question: 440

Examine the following code snippet and identify which statements regarding the use of *args and **kwargs in function definitions are true: (Select All That Apply)

def function_with_args(*args, **kwargs):
print(args)
print(kwargs)

function_with_args(1, 2, three='3', four='4')

A. The output will be (1, 2) and {'three': '3', 'four': '4'}
B. *args allows for variable numbers of positional arguments
C. **kwargs allows for variable numbers of keyword arguments
D. Both *args and **kwargs can be used in the same function

Answer: A, B, C, D

Explanation:
The *args parameter collects extra positional arguments into a tuple, while **kwargs collects additional keyword arguments into a dictionary. The output confirms that args contains (1, 2) and kwargs contains

{'three': '3', 'four': '4'}. Both can be used together in the same function definition.

Question: 441

What will be the output of the following code snippet?

```
def func(x):
return x * 2

print(func(func(3)))
```

A. 6
B. 3
C. 12
D. 9

Answer: C

Explanation:
The function func takes an argument x and returns x * 2.

When func(3) is called, it returns 6.
Then, func(6) is called, which returns 12.
Thus, the final output is 12.

Question: 442

How do you define a function that can take an arbitrary number of positional arguments and keyword arguments?

A. def my_function(*args, kwargs):
B. def my_function(args, kwargs):
C. def my_function(*args, **args):

Answer: A

Explanation:
The syntax *args captures positional arguments as a tuple, while **kwargs captures keyword arguments as a dictionary.

Question: 443

A healthcare application requires a Python program to calculate the Body Mass Index (BMI) from a user's weight and height. The program must handle exceptions for invalid input and provide a clear output. You need to complete the code to meet the requirements.

```
weight = XXX
height = YYY
try:
bmi = weight / (height ** 2)
except ZeroDivisionError:
print('Height cannot be zero.')
else:
print('Your BMI is: {:.2f}'.format(bmi))
```

What should you insert instead of XXX and YYY?

A. XXX -> float(input('Enter your weight in kg: ')) YYY -> float(input('Enter your height in meters: '))
B. XXX -> input('Enter your weight in kg: ') YYY -> input('Enter your height in meters: ')
C. XXX -> float(input('Enter your weight in kg: ')) YYY -> input('Enter your height in meters: ')
D. XXX -> input('Enter your weight in kg: ') YYY -> float(input('Enter your height in meters: '))

Answer: A

Explanation:

```
weight = float(input('Enter your weight in kg: '))
height = float(input('Enter your height in meters: '))
try:
bmi = weight / (height ** 2)
except ZeroDivisionError:
print('Height cannot be zero.')
else:
print('Your BMI is: {:.2f}'.format(bmi))
```

The program captures user input as a float for weight and height.
It handles division by zero, which can occur if height is zero, and formats the BMI output to two decimal places.

Question: 444

Which of the following statements correctly illustrates the concept of variable scope in Python?

A. Variables defined inside a function can be accessed from outside the function.
B. Global variables can be modified inside a function using the global keyword.
C. Local variables can be accessed from any part of the program.
D. The scope of a variable is determined by the order of its declaration.

Answer: B

Explanation:
In Python, variables defined inside a function are local to that function and cannot be accessed from outside. To modify a global variable inside a function, you must declare it using the global keyword.

Question: 445

What will be printed by the following code snippet when executed?

```
def func(x):
if x >= 0:
return x
else:
return func(-x)

print(func(-5))
```

A. 5
B. -5
C. 0

Answer: A

Explanation:
The function func checks if x is non-negative.

When func(-5) is called, -5 is negative, so it calls func(5).
Now, func(5) checks and finds that 5 is non-negative and returns 5.
Thus, the output will be 5.

Question: 446

When using keyword arguments in a function, which of the following is a valid way to call the function?

A. my_function(x=1, 2)
B. my_function(2, x=1)
C. my_function(2, y=3)

Answer: B

Explanation:
Keyword arguments can be specified in any order. However, positional arguments must appear before keyword arguments.

Question: 447

What does the following code output?

```
def is_palindrome(s):
return s == s[::-1]
print(is_palindrome("racecar"))
```

A. True
B. False
C. racecar
D. 1
E. 0

Answer: A

Explanation:

The function is_palindrome checks if the string s is the same forwards and backwards. "racecar" is a palindrome, so the output is True.

Question: 448

Consider the following code snippet. What will be the output when this code is executed?

```
def merge_dicts(dict1, dict2):
return {**dict1, **dict2}

dict_a = {'a': 1, 'b': 2}
dict_b = {'b': 3, 'c': 4}
print(merge_dicts(dict_a, dict_b))
```

A. {'a': 1, 'b': 2, 'c': 4}
B. {'a': 1, 'b': 3, 'c': 4}
C. {'a': 1, 'b': 2}
D. {'b': 3, 'c': 4}

Answer: B

Explanation:

The function merge_dicts combines two dictionaries. If there are overlapping keys, the values from the second dictionary (dict2) take precedence. Hence, the resulting dictionary is {'a': 1, 'b': 3, 'c': 4}.

KILLEXAMS.COM

Killexams.com is a leading online platform specializing in high-quality certification exam preparation. Offering a robust suite of tools, including Exam Questions, practice tests, and advanced test engines, Killexams.com empowers candidates to excel in their certification exams. Discover the key features that make Killexams.com the go-to choice for exam success.

Practice Exam Questions Based on Current Exam Objectives

Killexams.com provides practice exam questions aligned with the latest official exam objectives and latest syllabus. Our content is reviewed and updated regularly to reflect recent changes announced by certification vendors. By studying these practice questions, candidates will cover the structure, difficulty level, and topics of the actual exam, helping them prepare more effectively and efficiently.

Comprehensive Practice Exams (PDF Format)

Killexams.com offers multiple-choice questions (MCQs) in easy-to-read PDF format, covering all major domains of the exam. Each PDF contains a structured collection of practice questions and verified answers designed to support focused study. These MCQs help candidates reinforce key concepts, identify knowledge gaps, and improve exam readiness through consistent practice.

Realistic Practice Tests (Online Test Engine & Desktop Test Engine)

To support hands-on preparation, Killexams.com provides practice tests through both an Online Test Engine and a Desktop Test Engine. These tools are designed to simulate a real exam environment, allowing candidates to practice under exam-like conditions, with latest syllbus and topics of the exam. Performance tracking, test history, and result analysis help users evaluate their progress and focus on areas that need improvement.

Risk-Free Purchase Policy

Killexams.com follows a transparent and customer-friendly purchase policy. If users are not satisfied with the study materials, they may request assistance or a refund in accordance with our published terms and conditions. This policy reflects our commitment to customer satisfaction, fairness, and confidence in our preparation resources.

Regularly Updated Content

Our practice question bank is reviewed and updated on an ongoing basis to stay aligned with the latest exam outlines and vendor updates. This ensures candidates are studying up-to-date, relevant material, and preparing with content that reflects current exam expectations, helping them stay confident and well-prepared.