True/FalseQuestions

1. zip() modifies the original iterables that it combines.-False
2. zip() can only combine two iterables at a time- False
3. The resulting iterator from zip() contains elements up to the length of the longest input iterable.-False
4. Lazy objects in Python conserve memory as they calculate values only when requested.-True
5. map() applies a function to each item in an iterable and returns a list of the results-False
6. List comprehensions are eager and generate the entire list immediately when executed-True
7. List comprehensions are limited to iterating over only one iterable-False
8. List comprehensions allow the use of statements like if and else for conditional filtering and value assignment-True
9. In Python, everything is an object- True
10. Python supports method overloading, where multiple methods with the same name can have different parameters-False
11. Is Method overriding is possible between the class having composition Relationship-True
12. Types of Inheritance depend upon the number of child and parent classes involved- true
13. There can be multiple except statement in exception handling -True
14. Is it possible to create an empty class in Python- True
15. Inheritance in Python allows a class to inherit properties and methods from multiple parent classes.-True
16. The raise statement allows the programmer to force a specific exception to occur- true
17. Python has a garbage collector that handles memory management automatically-True
18. An except block in Python can handle multiple exceptions by specifying them within a single block separated by commas-True
19. The order of except blocks in Python matters; more specific exceptions should be listed before generic ones. True
20.finally block in Python is executed only when an exception occurs.-False

Fill in the blanks.

21	The return type of the zip() function is an ____**iterator**_____ object in Python.
22	The zip() function is commonly used to iterate over multiple iterables _ **simultaneously** _____
23	When applying zip() with iterables of different lengths, the resulting iterator will contain elements up to the length of the ____**short**_____ input iterable.
24	Using map() with a lambda function can be useful for applying simple operations or

	transformations to _____ **each element of an iterable** _____.
25	The filter() function in Python constructs an iterator from elements of an iterable for which a function returns _____**True**_____.
26	The map() function in Python applies a given function to every item in an iterable and returns a new iterator, effectively creating a __ **transformed version of original iterable**_____.
27	map() can take multiple iterables as arguments and apply a function _in **parallel**_____.
28	reduce() takes a function and an iterable as arguments, repeatedly applying the function to pairs of elements, _____.
29	**Data Abstraction** concept ensures access to only the implementation of a function and not its internal data and function.
30	Function that accepts multiple parameters but can have only one statement is_____**lambda**_____
31	**Constructor** Function is created by default when a object is created.
32	__init__() method in Python classes is used to **initialize the instance variable**_____.
33	A---**Static**-------method is a method that is bound by the class itself and not the object
34	Using --**super**----------, the child class can refer to any method of the parent class.
35	In _____**Multilevel**_____inheritance, features of the base class and the derived class are further inherited into the new derived class
36.	Inheritance consisting of multiple types of inheritance is called _____hybrid_____ inheritance
37.	The __Destructor_____ was called after the program ended or when all the references to object are deleted
38	Attributes are always ____public_____and can be accessed using the dot (.) operator
39.	___lambda_____ functions are quick, throwaway single line functions
40	Composition in Python refers to the design principle where a class contains objects of other

	classes as _____objects_____.
41	Method overriding occurs when a subclass provides a specific implementation of a method that is already defined in its ____Base class_____.
42	Instance methods in Python take the instance itself as the first parameter, typically referred to as ____Self_____.
43	Class attributes in Python are shared among all instances of the class, whereas instance attributes are specific to each __objects_____.
	Question
44	What is the purpose of the zip() function in Python? The main purposes of the zip() function include: Aggregating Elements: It allows to group corresponding elements from different iterables together into tuples. Iterating Simultaneously: It facilitates iterating over multiple iterables simultaneously in a concise and readable way.
45	Can you use zip() to combine more than two iterables? If so, how? Yes list1 = [1, 2, 3] list2 = ['a', 'b', 'c'] list3 = [10, 20, 30] list4 = ['x', 'y', 'z'] zipped = zip(list1, list2, list3, list4 for item in zipped: print(item)
46	How does zip() handle iterables of different lengths? The resulting iterator from zip() will only contain tuples up to the length of the shortest iterable provided. Ex list1 = [1, 2, 3] list2 = ['a', 'b'] list3 = ['x', 'y', 'z', 'w'] zipped = zip(list1, list2, list3) for item in zipped: print(item)
47	How can you unzip a zipped object created by zip()?

	list1 = [1, 2, 3] list2 = ['a', 'b', 'c'] zipped = zip(list1, list2) unzipped_list1, unzipped_list2 = zip(*zipped) print(list(unzipped_list1)) print(list(unzipped_list2)
48	Can you use map() with multiple iterables? If so, how? Yes def add_three_numbers(a, b, c): return a + b + c list1 = [1, 2, 3] list2 = [4, 5, 6] list3 = [7, 8, 9] result = map(add_three_numbers, list1, list2, list3) output = list(result) print(output)
49	Is filter() eager or lazy in its evaluation? In Python, the filter() function is lazy in its evaluation. The filter() function creates an iterator of elements for which a function returns True. However, filter() does not compute or execute the filtering immediately. Instead, it returns an iterator that yields elements from the input iterable based on the evaluation of the provided function when each element is requested. This makes it a lazy evaluation process. Lazy evaluation is beneficial in terms of memory efficiency because it doesn't compute all the values at once; it generates elements on-the-fly as required.
50	Is it possible to accomplish the same task using a loop instead of reduce()? Yes we can achieve a similar result as the reduce() function using a loop, this can be done with the following example . from functools import reduce numbers = [1, 2, 3, 4, 5] **Finding the sum of numbers using reduce()** result_reduce = reduce(lambda x, y: x + y, numbers) print(result_reduce) numbers = [1, 2, 3, 4, 5] **Finding the sum of numbers using a loop** sum_result = 0 for num in numbers: sum_result += num

	print(sum_result)
51	Can you use filter() with a lambda function? Show an example. The filter() function in Python can be used with a lambda function to filter elements from an iterable based on a specified condition. Here's an example demonstrating the usage of filter() with a lambda function: numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] filtered_numbers = list(filter(lambda x: x % 2 == 0, numbers))
52	Compare and contrast map, filter, and reduce. map() Function: Purpose: The map() function applies a specified function to each item in an iterable and returns an iterator that yields the results. Usage: It's useful when you want to transform each element of an iterable based on a given function. Syntax: map(function, iterable1, iterable2, ...) Example: map(lambda x: x * 2, [1, 2, 3, 4]) will return an iterator containing [2, 4, 6, 8]. filter() Function: Purpose: The filter() function constructs an iterator from elements of an iterable for which a function returns True. Usage: It's used to selectively extract elements from an iterable based on a specific condition. Syntax: filter(function, iterable) Example: filter(lambda x: x % 2 == 0, [1, 2, 3, 4, 5, 6]) will return an iterator containing [2, 4, 6]. reduce() Function: Purpose: The reduce() function (available in Python 2 and in the functools module in Python 3) performs a rolling computation on an iterable and returns a single value. Usage: It's used when you need to perform some computation that reduces an iterable to a single cumulative result. Syntax: reduce(function, iterable[, initializer]) Example: reduce(lambda x, y: x + y, [1, 2, 3, 4]) will return 10 (sum of all elements). Key Differences: Functionality: map() transforms each element based on a given function, filter() selects elements based on a condition, and reduce() aggregates elements to produce a single value. Result Type: map() returns an iterator of transformed elements, filter() returns an iterator of

	selected elements, and reduce() returns a single value. Arguments: map() and filter() take a function and one or more iterables, while reduce() takes a function and an iterable (and an optional initializer). Availability: map() and filter() are built-in functions, while reduce() is available in the functools module in Python 3 and is a built-in function in Python 2
53	Can you describe the role of the finally block when handling exceptions in Python?
54	What is the syntax for List comprehension in Python? new_list = [expression for item in iterable if condition] Explanation of the components: expression: The operation or expression that will be applied to each element in the iterable to generate the elements of the new list. item: A variable that represents each element in the iterable. iterable: The original iterable (list, tuple, set, etc.) from which elements are taken. if condition (optional): A condition that filters elements from the iterable based on this condition. It is optional to include this part.
55	How are map calls and list comprehensions related? Compare and contrast the two.
56	Explain the syntax of handling multiple exceptions using separate except blocks.
57	Discuss inheritance in Python programming language. Write a Python program to demonstrate the use of super() function
58	Define class.
59	Define Inheritance
60	Define Polymorphism
61	What do you think are the merits of object-oriented programming?
62	Explain constructors and Destructors
63	How does Python support multiple inheritances? What are the potential issues and how can they be mitigated?
64	What is composition in Python? How does it differ from inheritance?
65	Explain the concept of method resolution order (MRO) in Python multiple inheritance.
	Programming Question
66	Write a Python program to filter a list of integers using Lambda. Original list of integers: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] Even numbers from the said list: [2, 4, 6, 8, 10] Odd numbers from the said list: [1, 3, 5, 7, 9] original_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] even_numbers = list(filter(lambda x: x % 2 == 0, original_list)) odd_numbers = list(filter(lambda x: x % 2 != 0, original_list)) print("Original list of integers:") print(original_list) print("Even numbers from the said list:") print(even_numbers) print("Odd numbers from the said list:")

	print(odd_numbers)
67	Write a Python program to convert all the characters into uppercase and eliminate duplicate letters from a given sequence. Use the map() function. Input: hello world Output: HWRD LOE sequence = "hello world" result = ''.join(set(map(lambda x: x.upper(), sequence))) print("Original Sequence:", sequence) print("Sequence with characters in uppercase and duplicates removed:",result)
68	Write a Python program that prompts the user to input an integer and raises a ValueError exception if the input is not a valid integer. try: user_input = input("Please enter an integer: ") user_integer = int(user_input) print("Input is a valid integer:", user_integer) except ValueError: print("ValueError: Please enter a valid integer.")
69	Write a Python program that opens a file and handles a FileNotFoundError exception if the file does not exist. file_name = "file.txt" try: with open(file_name, 'r') as file: print("File opened successfully.") except FileNotFoundError: print("Error: File not found.")
70	Write a Python program that executes an operation on a list and handles an IndexError exception if the index is out of range try: my_list = [1, 2, 3] index = 5 value = my_list[index] print(f"The value at index {index} is: {value}") except IndexError: print(f"Error: Index {index} is out of range.")

71	Write a Python program to create a person class. Include attributes like name, country and date of birth. Implement a method to determine the person's age named as calculate_age() that calculates the person's age based on the current date and their date of birth.
	
	from datetime import date
	
	class Person:
	def __init__(self, name, country, date_of_birth):
	self.name = name
	self.country = country

```
        self.date_of_birth = date_of_birth

    def calculate_age(self):
        today = date.today()
        birth_date = self.date_of_birth
        age = today.year - birth_date.year - ((today.month, today.day) < (birth_date.month, birth_date.day))
        return age

person1 = Person("John", "USA", date(1990, 5, 15))
age = person1.calculate_age()
print(f"{person1.name} is {age} years old.")
```

72 Write a Python program to create a class that represents a shape. Include methods to calculate its area and perimeter. Implement subclasses for different shapes like circle, triangle, and square.

```
import math
class Shape:
    def area(self):
        pass

    def perimeter(self):
        pass

class Circle(Shape):
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return math.pi * self.radius**2

    def perimeter(self):
        return 2 * math.pi * self.radius

class Triangle(Shape):
    def __init__(self, side1, side2, side3):
        self.side1 = side1
        self.side2 = side2
        self.side3 = side3

    def perimeter(self):
        return self.side1 + self.side2 + self.side3

    def area(self):
        s = (self.side1 + self.side2 + self.side3) / 2
        return math.sqrt(s * (s - self.side1) * (s - self.side2) * (s - self.side3))
```

```
class Square(Shape):
    def __init__(self, side):
        self.side = side

    def area(self):
        return self.side**2

    def perimeter(self):
        return 4 * self.side

circle = Circle(5)
print("Circle - Area:", circle.area())
print("Circle - Perimeter:", circle.perimeter())

triangle = Triangle(3, 4, 5)
print("\nTriangle - Area:", triangle.area())
print("Triangle - Perimeter:", triangle.perimeter())

square = Square(6)
print("\nSquare - Area:", square.area())
print("Square - Perimeter:", square.perimeter())
```

73 Write a Python program to create a class representing a shopping cart. Include methods for adding and removing items, and calculating the total price.

74 Given the Coordinates (x, y) of a center of a Circle and its radius, write Python program to determine whether the Point lies inside the Circle, on the Circle or outside the Circle

```
import math

class Circle:
    def __init__(self, center_x, center_y, radius):
        self.center_x = center_x
        self.center_y = center_y
        self.radius = radius

    def point_position(self, x, y):
        distance = math.sqrt((x - self.center_x) ** 2 + (y - self.center_y) ** 2)

        if distance < self.radius:
            return "Inside the circle"
        elif distance == self.radius:
            return "On the circle"
        else:
            return "Outside the circle"

circle = Circle(0, 0, 5)
point_x = 3
point_y = 4
position = circle.point_position(point_x, point_y)
```

	print(f"The point ({point_x}, {point_y}) is {position} of the circle.")
75	Create a Bus child class that inherits from the Vehicle class. The default fare charge of any vehicle is seating capacity * 100. If Vehicle is Bus instance, we need to add an extra 10% on full fare as a maintenance charge. So total fare for bus instance will become the final amount = total fare + 10% of the total fare.
76	Let a be the list of values produced by range(1, 11). Using the function filter and a lambda argument, write an expression that will produce each of the following: a. A list of the even values in a b. A list of the values in a divisible by 3
77	Explain the following execution of the function filter .Hint: remember how integer values are interpreted when a Boolean is required. >>>filter(lambda x:x,[4,0,6,3,0,2]) [4,6,3,2]Write a lambda function for each of the following: a. Take one argument and return true if it is nonzero b. Take one argument and return true if it is odd c. Take two arguments, and return their sum d. Take two arguments, and return true if their sum is odd e. Take three arguments, and return true if the produce of the first two is less than or equal to the third
78	Let a be the list of values produced by range(1, 11). Using the function map and a lambda argument, write an expression that will produce each of the following a. A list of squares of the values b. A list of cubes of the values c. A list where each element is larger by one than the corresponding element in the original list
79	Write a function named Square that returns the squares of all the numbers of a list argument passed to it in sorted order from lowest to highest. Your code must make use of list comprehensions
	Using list comprehension print the Fibonacci Sequence in comma separated form for given input n.
80	Using list comprehension ,write a program to print the list with numbers which are divisible by 5 and 7 in [12,24,35,70,88,120,155].
81	For class FullName that contains an __init__ method that is passed a first and last name to construct a new object with, define a subclass of the FullName class named FullNamePlus that also stores a person's middle name. The subclass should include an __init__ method, getter/setter methods, and a __str__ methods that displays a name in the form lastname, firstname middle name. (Assume that class Full Name contains getter methods accessing the first and last names.)
82	Give the definition of an abstract class named Currency that stores a currency amount (as an integer value). Include in the class an __init__ method and an abstract __str__ method. Define three subclasses of the Currency class named US Currency (dollars), Euro Currency (euros), and the Chinese Currency (yuan). Implement the __str__ method of each so that the proper currency symbol is displayed. The Unicode string for displaying Euro and Chinese

	currency symbols are '\u20AC' and '\u5143, respectively.
83	Give section of code that prompts the user for an integer value, and displays the number entered if no exception is raised, otherwise catches the exception when a non-numeric character is entered, displaying the message "non-digit found in input."
84	Write a function called getYearOfBirth that prompts the user to enter their year of birth as a four-digit number, and returns the year entered. The function should throw a ValueError exception if the entered number is not within the range 1900 to the present year. Write a program that calls this function, and keeps calling it until a valid year has been entered.
85	Give appropriate getter and setter methods for an XYCoord class containing private instance variables x and y.Give an appropriate __init__ method for a class named FullName that stores the first and last name of a given individual. Assume that the class contains (private) instance variables first_name and last_name.
86	For person1, an object instance of classFullName containing private instance variables first_name and last_name, indicate how each of these instance variables can be directly accessed from outside the class.
87	For a class named FullName that contains an __init__ method that is passed a first and last name to construct a new object with, give an instruction that creates a new object of type FullName containing the name "Yu Huang."
