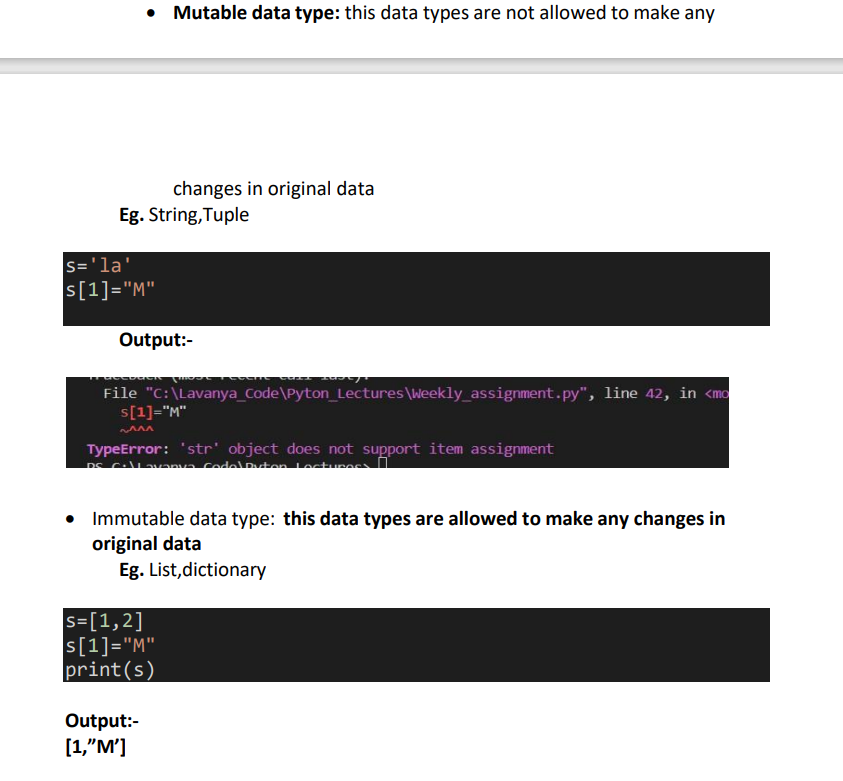
# Section A: Theory (10 Questions)

# 1. Explain the difference between mutable and immutable data types with examples.



# 2. What are nested loops? Describe a real-life scenario where nested loops can be used effectively.

# 3. Define a lambda function. How is it different from a normal function?

# 4. Explain the concept of scope in Python with examples of local and global variables.

# 5. What is exception handling? Why is it important in programming?

# 6. Discuss the use of the map, reduce, and filter functions in Python. Provide a brief example for each.

# 7. How do you slice a list in Python? Provide an example demonstrating slicing with a step value.

# 8. What is the difference between if-else and nested if-else statements? Provide an example for both.

# 9. Describe how to import and use modules in Python. Provide an example of importing a module and accessing its functions.

# 10. Explain the use of try-except blocks. How do they help in handling errors in Python programs?

# Section B: Correct the Code (10 Questions)

# 1.

# num = input("Enter a number: ")

# if num % 2 == 0:

#     print("Even number")

# else:

#     print("Odd number")

# ANS:

# num = int(input("Enter a number: "))

# if num % 2 == 0:

#     print("Even number")

# else:

#     print("Odd number")

# 2.

# my\_list = [1, 2, 3, 4]

# for i in range(5):

#     print(my\_list[i])

# ANS:

# my\_list = [1, 2, 3, 4]

# for i in range(len(my\_list)):

#     print(my\_list[i])

# 3.

# def add\_numbers(a, b):

#     result = a + b

# print(result)

# ANS:

# result=0

# def add\_numbers(a, b):

#     result = a + b

#     print(result)

# add\_numbers(10,20)

# 4.

# numbers = [1, 2, 3, 4, 5]

# squared = map(lambda x: x \*\* 2, numbers)

# print(squared)

# ANS:

# numbers = [1, 2, 3, 4, 5]

# squared = list(map(lambda x: x \*\* 2, numbers))

# print(squared)

# 5.

# def calculate\_area(radius):

#     return 3.14 \* radius \*\* 2

# area = calculate\_area()

# print(area)

# def calculate\_area(radius):

#     return 3.14 \* radius \*\* 2

# area = calculate\_area(2)

# print(area)

# 6.

# data = [10, 20, 30, 40]

# for value in data:

#     if value > 20:

#         print("Value: ", value)

#     else:

#         break

# ANS:

# data = [10, 20, 30, 40]

# for value in data:

#     if value > 20:

#         print("Value: ", value)

#     else:

#         pass

# 7.

# try:

#     num1 = 10

#     num2 = 0

#     result = num1 / num2

# except:

#     print("Error")

# ANS:

# try:

#     num1 = 10

#     num2 = 0

#     result = num1 / num2

# except:

#     print("Zero divistion Error")

# 8.

# text = "Hello World"

# print(text[11])

# ANS:

# text = "Hello World"

# print(text[10])

# 9.

# result = reduce(lambda x, y: x + y, [1, 2, 3, 4, 5])

# print("Sum: " + result)

# from functools import reduce

# l=[1,2,3,4,5]

# def add(x,y):

#     return x+y

# r=reduce(add,l)

# print(r)

# 10.

# def add(a, b, c):

#     return a + b + c

# result = add(5, 10)

# print("Result: ", result)

# ANS:

# def add(a, b=0, c=0):

#     return a + b + c

# result = add(5, 10)

# print("Result: ", result)

# Section C: Write Code For (10 Questions)

# 1. Create a program that accepts a string input from the user and prints the string in

# reverse order using slicing.

# s=input("Enter String")

# print("Reverse String: ",s[::-1])

# output:

# Enter Stringlavanya

# Reverse String:  aynaval

# 2. Write a program that calculates the factorial of a number using a loop.

# n=int(input("enter no "))

# fact=1

# for i in range(1,n+1):

#     fact\*=i

# print(fact)

# output:

# enter no 5

# 120

# 3. Create a function that takes two numbers and checks if they are divisible by each other.

# Return a message accordingly.

# n1=int(input("Enter no1 "))

# n2=int(input("Enter no2 "))

# if n1%n2==0 and n2%n1==0:

#     print("Both number devisible by each other ")

# else:

#     print("Both number are not divisible by each other")

# output:

# Enter no1 40

# Enter no2 40

# Both number devisible by each other

# 4. You are given a nested list of integers. Write a program to flatten the list

# and calculate the sum of all numbers in the flattened list.

# l=[10,20,30,[20,30],[20]]

# lst=[]

# for i in l:

#     if type(i)==list:

#         for j in i:

#             lst.append(j)

#     else:

#         lst.append(i)

# print(sum(lst))

# output:

# 130

# 5. Write a program to find the second largest number in a list without sorting it.

# from copy import copy

# l=[1,2,3,40,50,60,70]

# l2=copy(l)

# l2.remove(max(l))

# print("Secound\_Highest",max(l2))

# output:Secound\_Highest 60

# 6. Write a function that accepts a list of strings and returns the longest string

# using the reduce function.

# l=['lavanya','i','am','painte']

# from functools import reduce

# r=reduce(lambda x,y:x if len(x)>len(y) else y,l)

# print(r)

# output:lavanya

# 7. Your task is to create a program that prints a pyramid pattern of stars based on user input.

# The input specifies the number of levels in the pyramid.

# n=int(input("Enter no: "))

# for i in range(1,n+1):

#     for p in range(n-i):

#         print(" ",end="")

#     for k in range(i\*2-1):

#         print("\*",end="")

#     print()

# 8. A fictional app collects user age data. Write a program to categorize ages into groups

# # (child, teen, adult, senior) using a nested conditional.

# age=int(input("Enter age "))

# if age<=12:

#     print("Child")

# elif age in range(13,20):

#     print("Teen")

# elif age in range(20,60):

#     print("adult")

# elif age>65:

#     print("Senior")

# output:

# Enter age 8

# Child

# Enter age 67

# Senior

# Enter age 25

# adult

# 9. You are building a simulation for a bakery. The program should prompt the user to

# enter the type of pastry and the quantity. Based on the input, calculate the total cost

# and display it. Use a dictionary for pricing and nested loops for input validation.

# d={}

# n=int(input("Enter no of items you want "))

# for i in range(n):

#    type=input("Enter type of pastry ")

#    qty=int(input("Enter qty "))

#    d.update({type:qty})

# print(d)

# sum=0

# for k,v in d.items():

#     if k=='pinapple':

#         sum+=v\*50

#     elif k=='chacolava':

#         sum+=v\*100

# print(sum)

# output

# Enter no of items you want 2

# Enter type of pastry pinapple

# Enter qty 2

# Enter type of pastry chacolava

# Enter qty 1

# {'pinapple': 2, 'chacolava': 1}

# 200

# 10. A treasure map is represented as a 5x5 grid where 'X' marks a treasure and 'O' represents empty spots.

# Write a program that takes user input to navigate the grid and display the status after each move.

# Make sure to include boundary conditions and invalid input handling.

# map=[

# ['O','O','O','O','O'],

# ['O','O','O','O','O'],

# ['O','O','X','O','O'],

# ['O','O','O','O','O'],

# ['O','O','O','O','O']]

# print(map)

# r,c=1,1

# def validMove(r,c):

#     if r<0 or r>5 or c<0 or c>5:

#         return False

#     else:

#         return True

# def move(map,direction):

#     global r,c

#     if direction=='up':

#             r=r-1

#     elif direction=='down':

#             r=r+1

#     elif direction=='left':

#             c=c-1

#     elif direction=='right':

#             c=c+1

#     for i in map:

#         for j in i:

#             print(j,end=" ")

#         print()

#     if  not validMove(r,c):

#         print("Invalid Valid")

#         return

#     if map[r][c]=='X':

#         return "Congraculations you found Treager"

# while(True):

#     direction=input(("Tresure not foun select move "))

#     p=move(map,direction)

#     if p:

#         print(p)

#         break

#output:

# your at 1,1 position

# Tresure not foun select move down

# O O O O O

# O O O O O

# O O O O O

# O O X O O

# O O O O O

# O O O O O

# Tresure not foun select move right

# O O O O O

# O O O O O

# O O X O O

# O O O O O

# O O O O O

# Congraculations you found Treager