PYTHON WORKSHEET 3--

ARMAAN TAYAL (1024230087)

# Question 1:

def calculate\_diff(num):

if num > 17:

return (num - 17) \* 2

else:

return abs(num - 17)

print("For 22, the result is:", calculate\_diff(22))

print("For 14, the result is:", calculate\_diff(14))

# ---

# Question 2:

def check\_range(num):

return (num in range(100, 1001) or num in range(1000, 2001))

print("Is 150 in range?", check\_range(150))

print("Is 1500 in range?", check\_range(1500))

print("Is 50 in range?", check\_range(50))

# ---

# Question 3:

def flip\_string(s):

reversed\_str = ""

for char in s:

reversed\_str = char + reversed\_str

return reversed\_str

original = "robut"

flipped = flip\_string(original)

print("The reversed string is:", flipped)

# ---

# Question 4:

def count\_cases(s):

counts = {'upper': 0, 'lower': 0}

for char in s:

if 'a' <= char <= 'z':

counts['lower'] += 1

elif 'A' <= char <= 'Z':

counts['upper'] += 1

return counts['upper'], counts['lower']

text = "Hello World"

upper\_case\_count, lower\_case\_count = count\_cases(text)

print("Original String:", text)

print("No. of uppercase characters:", upper\_case\_count)

print("No. of lowercase characters:", lower\_case\_count)

# ---

# Question 5:

def get\_uniques(input\_list):

return list(set(input\_list))

sample\_list\_1 = [1, 2, 3, 3, 4, 4, 5, 6, 6]

unique\_items = get\_uniques(sample\_list\_1)

print("Original list:", sample\_list\_1)

print("List with distinct elements:", unique\_items)

# ---

# Question 6:

def filter\_even(input\_list):

return [num for num in input\_list if num % 2 == 0]

sample\_list\_2 = [1, 2, 3, 4, 5, 6, 7, 8, 9]

even\_nums = filter\_even(sample\_list\_2)

print("Original list:", sample\_list\_2)

print("Even numbers:", even\_nums)

# ---

# Question 7:

def create\_closure(x):

def inner\_add(y):

return x + y

return inner\_add

add\_five = create\_closure(10)

result = add\_five(5)

print("The result of the inner function is:", result)

# ---

# Question 8:

def show\_args(\*args):

print("The arguments are:")

for arg in args:

print(arg)

show\_args("Python", 2024, "Coding")

# ---

# Question 9:

def move\_pos(x, y, direction):

if direction == "up":

y += 1

elif direction == "down":

y -= 1

elif direction == "left":

x -= 1

elif direction == "right":

x += 1

return (x, y)

x\_coord, y\_coord = 0, 0

new\_coords = move\_pos(x\_coord, y\_coord, "up")

print("New position of the robot:", new\_coords)

# ---

# Question 10:

def classify\_temp(temp):

if temp < 15:

return "Moderate"

elif temp <= 30:

return "Hot"

else:

return "Very Hot"

print("Classifying 10:", classify\_temp(10))

print("Classifying 25:", classify\_temp(25))

print("Classifying 35:", classify\_temp(35))

# ---

# Question 11:

def is\_final\_position\_reached(path):

x, y = 0, 0

for move in path:

if move == "up":

y += 1

elif move == "down":

y -= 1

elif move == "left":

x -= 1

elif move == "right":

x += 1

return x == 2 and y == 0

path\_list = ["up", "up", "down", "down", "right", "right"]

print("Is the final position (2,0)?", is\_final\_position\_reached(path\_list))

# ---

# Question 12:

class Student\_Data:

def \_\_init\_\_(self, s\_id, s\_name):

self.student\_id = s\_id

self.student\_name = s\_name

self.student\_class = "Default Class"

def show\_info(self):

print("Student ID:", self.student\_id)

print("Student Name:", self.student\_name)

print("Student Class:", self.student\_class)

student\_a = Student\_Data(101, "Alice")

student\_a.show\_info()

# ---

# Question 13:

class Student\_Info:

def \_\_init\_\_(self, s\_id, s\_name):

self.id\_num = s\_id

self.full\_name = s\_name

student\_one = Student\_Info(101, "Bob")

student\_two = Student\_Info(102, "Charlie")

print("Attributes of student1:")

print("Student ID:", student\_one.id\_num)

print("Student Name:", student\_one.full\_name)

print("\nAttributes of student2:")

print("Student ID:", student\_two.id\_num)

print("Student Name:", student\_two.full\_name)

# ---

# Question 14:

import math

class CircleCalculations:

def \_\_init\_\_(self, rad):

self.radius = rad

def calculate\_area(self):

return math.pi \* self.radius \*\* 2

def calculate\_perimeter(self):

return 2 \* math.pi \* self.radius

my\_circle = CircleCalculations(5)

print("Area of the circle:", my\_circle.calculate\_area())

print("Perimeter of the circle:", my\_circle.calculate\_perimeter())

# ---

# Question 15:

class StringProcessor:

def \_\_init\_\_(self, text):

self.data = text

def get\_data(self):

return self.data

def show\_data(self):

print("The string is:", self.data)

def to\_upper(self):

return self.data.upper()

processor = StringProcessor("hello world")

print("Retrieved string:", processor.get\_data())

processor.show\_data()

print("Uppercase string:", processor.to\_upper())

# ---

# Question 16:

class Robot\_Status:

def \_\_init\_\_(self, name, task, level):

self.name = name

self.task = task

self.battery = level

def perform\_task(self):

self.battery -= 10

print("The robot is performing the task and the battery decreases by 10%.")

return self.battery

def recharge(self):

self.battery = 100

print("The robot is recharged and the battery is back to 100%.")

return self.battery

def get\_status(self):

print("Robot Name:", self.name)

print("Current Task:", self.task)

print("Battery