SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

K. J. Somaiya School of Engineering, Mumbai-77
(Somaiya Vidyavihar University)

Somaiya TRUST

Course:	Python Programming			**Semester:**	I	**Date:**	27/11/2024
Division:		**Batch:**		**SET**	A	**Name:**	
Exam:	OST			**Time:**	04:00 to 05:30 PM	**Roll No:**	

Q 1	Attempt Any ONE [Show all test Cases in Output]	Marks
1	Write a Python program that filters a list of integers and counts how many numbers are less than 10 and odd.	08

Test Case 1:	**Test Case 2:**	**Test Case 3:**
Sample List: [2, 7, 15, 22, 10, 6] Expected Result: [7], 1	Sample List: [1, 6, 15, -2, -17, 6] Expected Result: [1, -17], 2	Sample List: [17, -6, 15, -2, 17, 6] Expected Result: [], 0

Initialization of empty list & Input - 2 marks
Use of For loop - 2 marks
Use of If with both conditions - 2 mark
Count numbers - 1 mark
Print Statement & all test case - 1 marks

```
# Sample list of integers
numbers = [1, 4, 7, 9, 12, 15, 2, 8, 3, 11, 5, 6]
filtered_numbers = []

# Iterate through each number in the list
for num in numbers:
    if num < 10 and num % 2 != 0:
        filtered_numbers.append(num)

count = len(filtered_numbers1)

# Output the result
print(filtered_numbers1, count)
```

2	Write a python program that finds ***even numbers divisible by 5*** from 1 to 50 and appends them in a text file with the statement: "The 1 number is: " and so on. **[Attach Screenshot of Output File also]** **Expected Output:** The 1 number is 10 The 2 number is 20 The 3 number is 30 The 4 number is 40 The 5 number is 50	08

Initialization of filename & empty List - 1 marks
Logic to find filtered number - 3 marks
Open a file - 1 marks
Write a output in file - 3 marks

```
# File name to store the output
filename = "file.txt"

# Initialize an empty list to hold the numbers
even_numbers_divisible_by_5 = []

# Loop from 1 to 50 to find even numbers divisible by 5
for num in range(1, 51):
    if num % 2 == 0 and num % 5 == 0:
        even_numbers_divisible_by_5.append(num)

# Write to file with the specified format
with open(filename, 'w') as file:
    for index, number in enumerate(even_numbers_divisible_by_5, start=1):
        file.write(f"The {index} number is {number}\n")
```

Q 2	Attempt Any ONE	Marks
1	Write a Python function ***student_grades*** that takes a dictionary where the keys are student names and the values are lists of their scores in different subjects. The function should return a dictionary where each student's name is a key, and the value is a dictionary with their maximum, minimum and average score rounded to two decimal places.	12

Test Case 1:	**Test Case 1:**
grades = { 'Sameer': [85, 90, 78],, 'Vihan': [92, 88, 84], 'Kabir': [72, 75, 80] } **Output:** {'Sameer': {'Average': 84.33, 'Highest Score': 90, 'Lowest Score': 78}, 'Vihan': {'Average': 88.0, 'Highest Score': 92, 'Lowest Score': 84}, 'Kabir': {'Average': 75.67, 'Highest Score': 80, 'Lowest Score': 72}}	grades = { 'Sat': [85, 90, 78 , 45, 60], 'Chid': [92, 88], 'Anand': [72, 75, 80, 32] } **Output:** {'Sat': {'Average': 71.6, 'Highest Score': 90, 'Lowest Score': 45}, 'Chid': {'Average': 90.0, 'Highest Score': 92, 'Lowest Score': 88}, 'Anand': {'Average': 64.75, 'Highest Score': 80, 'Lowest Score': 32}}

Define function and return value - 2 marks
Logic to find avg, max & min - 4 marks
Store the result in dict - 2 marks
Calling function - 2 marks
Print & all testcase - 2 marks

```
def student_grades(grades):
    # Initialize an empty dictionary to store the results
    detailed_grades = {}

    # Iterate through the dictionary of grades
    for student, scores in grades.items():
        # Calculate the average for each student and round it to two decimal places
        average = round(sum(scores) / len(scores), 2)
        # Find the highest and lowest scores
        highest_score = max(scores)
        lowest_score = min(scores)

        # Store the average, highest, and lowest scores in a dictionary for each student
        detailed_grades[student] = {
            'Average': average,
            'Highest Score': highest_score,
            'Lowest Score': lowest_score
        }

    return detailed_grades

# Sample input
grades = { 'Sameer': [85, 90, 78],,
    'Vihan': [92, 88, 84],
    'Kabir': [72, 75, 80] }

# Call the function and print the output
print(student_grades(grades))
```

2	Write a Python program that performs the following tasks: **1. Generate Data:** ● Use NumPy to create a range of values (x) from 0 to 4π (approximately 12.566) with an interval of 0.2. ● Compute the following trigonometric values for each x: ○ Sine values and store them in an array *y_sin*. ○ Cosine values and store them in an array *y_cos*. ○ Tangent values and store them in an array *y_tan*.	12

2. Plotting:

- Create a line plot that displays the sine, cosine, and tangent functions on the same graph.
- Give separate colors to each line
- Give the appropriate labels, title.
- Add grid lines & legend to the plot for better readability.

Generate Data using numpy - 2 marks
Compute all trigonometric values using numpy - 3 marks
Plot the graph - 3 marks
Use of different color - 1 marks
Labels & Titles - 1 mark
Grid & Legend - 1 mark
Output - 1 mark

```
import numpy as np
import matplotlib.pyplot as plt

# Step 1: Generate Data
# Create a range of x values from 0 to 4π (approximately 12.566) with an interval of 0.1
x = np.arange(0, 4 * np.pi, 0.2)

# Compute trigonometric values
y_sin = np.sin(x)  # Sine values
y_cos = np.cos(x)  # Cosine values
y_tan = np.tan(x)  # Tangent values

# Step 2: Plotting
# Create a line plot for the sine, cosine, and tangent functions
plt.figure(figsize=(10, 6))
plt.plot(x, y_sin, label='Sine', color='blue')   # Sine plot with label
plt.plot(x, y_cos, label='Cosine', color='green')  # Cosine plot with label
plt.plot(x, y_tan, label='Tangent', color='red')  # Tangent plot with label

# Label the axes
plt.xlabel('X values (radians)')
plt.ylabel('Y values')

# Add a title to the plot
plt.title('Trigonometric Functions: Sine, Cosine, and Tangent')

# Add grid lines for better readability
plt.grid(True)
```

```
# Add a legend to differentiate between the curves
plt.legend()

# Limit y-axis to avoid extreme values for tangent function (optional)
plt.ylim(-10, 10)

# Display the plot
plt.show()
```

SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

K. J. Somaiya School of Engineering, Mumbai-77
(Somaiya Vidyavihar University)

Somaiya TRUST

Course:	Python Programming			**Semester:**	I	**Date:**	27/11/2024
Division:		**Batch:**		**SET**	B	**Name:**	
Exam:	OST			**Time:**	04:00 to 05:30 PM	**Roll No:**	

Q 1 — **Attempt Any ONE [Show all test Cases in Output]** — **Marks**

1 — 08

Write a python to find out whether letter **'C'** OR letter **'D'** is available or not in a given string and count how many times that letter is present. (case-insensitive)

Test Case 1:	**Test Case 2:**	**Test Case 3:**
Input: "KJSCE" Output: True and Count = 1	Input: "Code" Output: True and Count = 2	Input: "Python" Output: False and Count = 0

Initialization of counter & Input - 2 marks
Use of For loop - 2 marks
Use of If & handle case insensitive - 2 mark
Print Statement & all test case - 2 marks

```
# Input string
input_string = input("Enter a string: ")
# Convert the string to uppercase for case-insensitive comparison
input_string = input_string.upper()

# Initialize the total count for letters 'C' and 'D'
total_count = 0

# Iterate through each character in the string and count 'C' or 'D'
for char in input_string:
    if char == 'C' or char == 'D':
        total_count += 1

# Determine if 'C' or 'D' is present
if total_count > 0:
    print(f"Output: True and Count = {total_count}")
else:
    print(f"Output: False and Count = {total_count}")
```

2 — 08

Write a program that reads data from a file and counts the number of vowels and consonants in the file. **[Attach Screenshot of Input File also]**

File Data: "Hello, today is Monday" **Output:** Count of vowels: 7 Count of consonants: 11	**File Data:** "Dear, I have $25" **Output:** Count of vowels: 5 Count of consonants: 4

Initialization of filename & Counter - 1 marks
Open & read a file - 2 marks
Logic to find vowels & Consonant - 3 marks
Print result & all test case - 2 marks

```
vowels = "aeiouAEIOU"
consonants = "bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ"
vowel_count = 0
consonant_count = 0

filename = "example.txt"  # Replace with the actual file name

with open(filename, 'r') as file:
    content = file.read()  # Read the entire content of the file

    for char in content:
        if char in vowels:
            vowel_count += 1
        elif char in consonants:
            consonant_count += 1

# Output the result
print(f"Count of vowels: {vowel_count}")
print(f"Count of consonants: {consonant_count}")
```

Q 2	Attempt Any ONE	Marks
1	Write a Python function ***login(username, password)*** that validates a user's credentials based on the following criteria: 1. **Username**: ○ Must be alphanumeric (only letters and numbers allowed). ○ Should have a maximum length of 12 characters. ○ Must contain at least two numeric digits. 2. **Password**: ○ Must be at least 8 characters long. ○ Should contain at least one uppercase letter. ○ Should contain at least one lowercase letter. If the username or password does not meet these criteria, the function should display **"Invalid username or password"**. If both are valid, display **"Login successful"**.	12

username: **User123** password: **Pass@123** Output: **Login successful**	username: **User1** password: **Pass@123** Output: **Invalid username or password**	username: **Username12345** password: **Pass@123** Output: **Invalid username or password**	username: **Username1234** password: **pass@123** Output: **Invalid username or password**

Define function - 1 marks
Logic to check username - 2+2 marks
Logic to check password - 2+2 marks
Calling function - 1 marks
Print & all testcase - 2 marks

```
def login(username, password):
    # Check if the username is alphanumeric and of the correct length
    if not (username.isalnum() and len(username) <= 12):
        print("Invalid username or password")
        return

    # Check if the username contains at least two numeric digits
    digit_count = 0
    for char in username:
        if char.isdigit():
            digit_count += 1
    if digit_count < 2:
        print("Invalid username or password")
        return

    # Check if the password is at least 8 characters long
    if len(password) < 8:
        print("Invalid username or password")
        return

    # Check if the password has at least one uppercase and one lowercase letter
    has_upper = False
    has_lower = False
    for char in password:
        if char.isupper():
            has_upper = True
        if char.islower():
            has_lower = True
```

```
    if not (has_upper and has_lower):
        print("Invalid username or password")
        return

    # If all conditions are met, login is successful
    print("Login successful")

# Sample test cases
login("User123", "Pass@123")        # Output: Login successful
login("Username12345", "Pass@123")      # Output: Invalid username or password
login("User1234", "pass@123")      # Output: Invalid username or password
login("User1", "Pass@123")      # Output: Invalid username or password
```

2 | 12

Write a Python program that performs the following tasks:

1. **Create Data:**
   - Arrays of the last three years for temperatures are defined in 3x12 array shape.
   - Calculate average of last three years temperature month wise and display.
2. **Create Subplots:**
   - Create two subplots that share the same x-axis.
     - The first subplot should display a **bar chart** showing the average monthly temperatures.
     - The second subplot should display a **line chart** showing the monthly temperatures of all three years.
   - Add appropriate titles & labels for each subplot:
   - Use distinct colors for the bar chart and line chart
   - Add gridlines & legends to both subplots for better readability.

Sample Data:
[15, 16, 20, 25, 30, 35, 40, 39, 34, 28, 22, 18] # Year 1
[14, 17, 21, 26, 31, 36, 38, 37, 33, 27, 21, 17] # Year 2
[13, 15, 19, 24, 29, 34, 39, 38, 32, 26, 20, 16] # Year 3

Generate Data using numpy - 2 marks
Compute & print avg temp - 2 marks
Plot the graph - 3 marks
Use of subplot - 1 mark
Use of different color - 1 marks
Labels & Titles - 1 mark
Grid & Legend - 1 mark
Output - 1 mark

```
import numpy as np
import matplotlib.pyplot as plt

# Step 1: Create Data
temperatures = np.array([
    [15, 16, 20, 25, 30, 35, 40, 39, 34, 28, 22, 18],  # Year 1
    [14, 17, 21, 26, 31, 36, 38, 37, 33, 27, 21, 17],  # Year 2
    [13, 15, 19, 24, 29, 34, 39, 38, 32, 26, 20, 16]   # Year 3
])

# Calculate average monthly temperatures
average_monthly_temperatures = np.mean(temperatures, axis=0)

# Display the calculated data
print("Average Monthly Temperatures (°C):", average_monthly_temperatures)

# Step 2: Create Subplots
# Define the x-axis labels (months)
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

# Create a figure with two subplots sharing the same x-axis
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 10), sharex=True)

# First Subplot: Bar Chart for Average Monthly Temperatures
ax1.bar(months, average_monthly_temperatures, color='skyblue', label='Average
Temperature')
ax1.set_title('Average Monthly Temperatures (Last 3 Years)')
ax1.set_ylabel('Temperature (°C)')
ax1.grid(True)
ax1.legend()

# Second Subplot: Line Chart for Monthly Temperatures of All Three Years
ax2.plot(months, temperatures[0], marker='o', label='Year 1', color='blue')
ax2.plot(months, temperatures[1], marker='s', label='Year 2', color='green')
ax2.plot(months, temperatures[2], marker='^', label='Year 3', color='orange')
ax2.set_title('Monthly Temperatures for Last 3 Years')
ax2.set_xlabel('Months')
ax2.set_ylabel('Temperature (°C)')
ax2.grid(True)
ax2.legend()
plt.tight_layout()
plt.show()
```