WEEK 1

1. Write a Python program to check whether a given number is even or odd.

Algorithm:

Start

Read a number n

If n mod 2 == 0, print Even

Else, print Odd

Stop

```
num = int(input("Enter a number: "))
if num % 2 == 0:
    print("Even")
else:
    print("Odd")

Enter a number: 3
Odd
```

1. Write a Python program to check whether a number is positive, negative, or zero.

Algorithm:

Start

Read number n

If n > 0, print Positive

Else if n < 0, print Negative

Else print Zero

Stop

```
num = float(input("Enter a number: "))
if num > 0:
    print("Positive")
elif num < 0:
    print("Negative")
else:
    print("Zero")

Enter a number: +6
Positive
```

1. Write a Python program to find the largest among three numbers.

Algorithm:

Start

Read numbers a, b, c

If a ≥ b and a ≥ c, largest is a

Else if b ≥ c, largest is b

Else largest is c

Print largest

Stop

```
a = float(input("Enter a: "))
b = float(input("Enter b: "))
c = float(input("Enter c: "))

if a >= b and a >= c:
    print("Largest =", a)
elif b >= c:
    print("Largest =", b)
else:
    print("Largest =", c)
```

```
Enter a: 55
Enter b: 77
Enter c: 22
Largest = 77.0
```

1. Write a Python program to check whether a given number is a prime number.

Algorithm:

Start

Read number n

If n ≤ 1, print Not Prime and stop

For i from 2 to √n

If n mod i == 0, print Not Prime and stop

If no divisors found, print Prime

Stop

```
n = int(input("Enter a number: "))

if n <= 1:
    print("Not prime")
```

```
else:
    for i in range(2, int(n**0.5)+1):
        if n % i == 0:
            print("Not prime")
            break
    else:
        print("Prime")
```

```
Enter a number: 8
Not prime
```

Week - 2

1. Write a Python program to find the factorial of a number.

Algorithm:

Start

Read number n

Set fact = 1

Repeat from i = 1 to n

Multiply fact = fact × i

Print factorial

Stop

```
n = int(input("Enter number: "))
fact = 1
for i in range(1, n+1):
    fact *= i
print("Factorial =", fact)
```

```
Enter number: 5
Factorial = 120
```

1. Write a Python program to check whether a number is a palindrome.

Algorithm:

Start

Read number n

Reverse the number

If reversed number equals original, print Palindrome

Else print Not Palindrome

Stop

```
n = input("Enter a number: ")
if n == n[::-1]:
    print("Palindrome")
else:
    print("Not palindrome")

Enter a number: 9963699
Palindrome
```

1. Write a Python program to check whether a given string is a palindrome.

Algorithm:

Start

Read string s

Reverse the string

Compare original and reversed strings

If equal, print Palindrome

Else print Not Palindrome

Stop

```
s = input("Enter string: ")
if s == s[::-1]:
    print("Palindrome")
else:
    print("Not palindrome")

Enter string: POP
Palindrome
```

Week - 3

1. Write a Python program to print the Fibonacci series up to N terms.

Algorithm:

Start

Read number of terms n

Initialize a = 0, b = 1

Repeat n times

Print a

Update a = b, b = a + b

Stop

```
n = int(input("Enter N: "))
a, b = 0, 1
for i in range(n):
    print(a, end=" ")
    a, b = b, a+b

Enter N: 5
0 1 1 2 3
```

1. Write a Python program to find the sum of digits of a number.

Algorithm:

Start

Read number n

Initialize sum = 0

While n > 0

Add last digit to sum

Remove last digit

Print sum

Stop

```
n = input("Enter number: ")
sum_digits = sum(int(d) for d in n)
print("Sum =", sum_digits)

Enter number: 5
Sum = 5
```

1. Write a Python program to count vowels and consonants in a string.

Algorithm:

Start

Read string s

Initialize vowels = 0, consonants = 0

For each character in s

If character is vowel, increment vowels

Else if alphabet, increment consonants

Print counts

Stop

```
s = input("Enter string: ").lower()
vowels = "aeiou"
v = c = 0
for ch in s:
    if ch.isalpha():
        if ch in vowels:
            v += 1
        else:
            c += 1
print("Vowels:", v)
print("Consonants:", c)
```

```
Enter string: IUSDFHOWIJDVNJKSNIO
Vowels: 6
Consonants: 13
```

Week - 4

1. Write a Python program to reverse a string without using built-in functions.

Algorithm:

Start

Read string s

Initialize empty string rev

Traverse string from left to right

Add each character to beginning of rev

Print reversed string

Stop

```
s = input("Enter string: ")
rev = ""
for ch in s:
    rev = ch + rev
print("Reversed:", rev)
```

```
Enter string: JKBIH79879
Reversed: 97897HIBKJ
```

1. Write a Python program to count the occurrence of each character in a string.

Algorithm:

Start

Read string s

For each character in s

Count number of occurrences

Store and display frequency

Stop

```
s = input("Enter string: ")
count = {}
for ch in s:
    count[ch] = count.get(ch, 0) + 1
print(count)
```

```
Enter string: HHKJKSKQJSJIAJSIJQKXN
{'H': 2, 'K': 4, 'J': 5, 'S': 3, 'Q': 2, 'I': 2, 'A': 1, 'X': 1, 'N':
1}
```

1. Write a Python program to create a simple calculator using conditional statements.

Algorithm:

Start

Read two numbers a, b

Read operator

If operator is +, add

If -, subtract

If *, multiply

If /, divide

Display result

Stop

```
a = float(input("Enter a: "))
b = float(input("Enter b: "))
op = input("Enter operator (+ - * /): ")

if op == '+':
    print(a + b)
elif op == '-':
    print(a - b)
```

```
elif op == '*':
    print(a * b)
elif op == '/':
    if b == 0:
        print("Error: Division by zero is not allowed.")
    else:
        print(a / b)
else:
    print("Invalid operator")
```

```
Enter a: 56
Enter b: 0
Enter operator (+ - * /): /
Error: Division by zero is not allowed.
```

Week - 5

1. Write a Python program to implement a menu-driven calculator using a loop (repeat until the user exits).

Algorithm:

Start

Display menu

Read choice

If exit chosen, stop

Else read two numbers

Perform selected operation

Repeat until exit

Stop

```
while True:
    print("\n1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit")
    ch = int(input("Choice: "))
    if ch == 5: break
    a = float(input("A: "))
    b = float(input("B: "))
    if ch == 1: print(a+b)
    elif ch == 2: print(a-b)
    elif ch == 3: print(a*b)
    elif ch == 4:
        if b == 0:
            print("Error: Division by zero is not allowed.")
        else:
```

```
            print(a/b)
    else: print("Invalid")


1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 4
A: 66
B: 0
Error: Division by zero is not allowed.

1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 7
A: 6
B: 9
Invalid

1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 3
A: 7
B: 9
63.0

1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 2
A: 6
B: 9
-3.0

1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 1
A: 99
B: 99
198.0

1.Add 2.Subtract 3.Multiply 4.Divide 5.Exit
Choice: 5
```

1. Write a Python program to generate a multiplication table for a given number (loop until the user stops).

Algorithm:

Start

Read number n

For i = 1 to 10

Print n × i

Ask user to continue

Repeat or stop

Stop

```
while True:
    n = int(input("Enter number: "))
    for i in range(1, 11):
        print(n, "x", i, "=", n*i)
    if input("Continue? (y/n): ") == "n":
        break
```

```
Enter number: 7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
Continue? (y/n): Y
Enter number: 66
66 x 1 = 66
66 x 2 = 132
66 x 3 = 198
66 x 4 = 264
66 x 5 = 330
66 x 6 = 396
66 x 7 = 462
66 x 8 = 528
66 x 9 = 594
66 x 10 = 660
Continue? (y/n): N
Enter number: 6
6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54
6 x 10 = 60
Continue? (y/n): n
```

1. Write a Python program to print different patterns using loop concepts (e.g., star patterns, number patterns).

Algorithm:

Start

Read number of rows n

For each row from 1 to n

Print required pattern

Stop

```
n = int(input("Enter n: "))
for i in range(1, n+1):
    print("*" * i)
```

```
Enter n: 10
*
**
***
****
*****
******
*******
********
*********
**********
```

FUNCTION-BASED QUESTIONS

Week - 6

1. Write a Python function that takes a user's name and prints a greeting message.

Algorithm:

Start

Define function to accept name

Print greeting message

Call function

Stop

```
def greet(name):
    print("Hello", name)

greet(input("Enter name: "))
```

```
Enter name: bishwas
Hello bishwas
```

1. Write a Python function that accepts two numbers and returns their sum.

Algorithm:

Start

Define function with two parameters

Return sum

Call function

Print result

Stop

```
def add(a, b):
    return a + b

print(add(88, 60))
```

```
148
```

Week – 7

1. Write a Python recursive function to find the factorial of a number.

Algorithm:

Start

Define recursive function

If n == 0, return 1

Else return n × fact(n−1)

Call function

Print result

Stop

```
def fact(n):
    if n == 0: return 1
    return n * fact(n-1)

print(fact(int(input("Enter n: "))))
```

```
Enter n: 7
5040
```

1. Write a Python lambda function to check whether a number is even.

Algorithm:

Start

Define lambda expression

Check modulus by 2

Print result

Stop

```
is_even = lambda x: x % 2 == 0
print(is_even(int(input("Enter number: "))))

Enter number: 8
True
```

1. Write a Python program to calculate factorial using recursion with input validation.

Algorithm:

Start

Read input

Validate input

If invalid, display error

Else calculate factorial recursively

Print result

Stop

```
def fact(n):
    if n == 0: return 1
    return n * fact(n-1)

n = input("Enter number: ")
if not n.isdigit():
    print("Invalid input")
else:
    print("Factorial =", fact(int(n)))

Enter number: 9
Factorial = 362880
```

PROJECT / ADVANCED QUESTIONS

Week - 8

1. Write a Python program to create a Library Book Management System using functions.

Algorithm:

Start

Create empty book list

Display menu

Add / View / Remove books

Repeat until exit

Stop

```
library = []

def add_book():
    book = input("Book name: ")
    library.append(book)
    print("Added.")

def view_books():
    print("Books:", library)

def remove_book():
    book = input("Book to remove: ")
    if book in library:
        library.remove(book)
        print("Removed.")
    else:
        print("Not found.")

while True:
    print("\n1.Add 2.View 3.Remove 4.Exit")
    ch = int(input("Choice: "))
    if ch == 1: add_book()
    elif ch == 2: view_books()
    elif ch == 3: remove_book()
    else: break
```

```
1.Add 2.View 3.Remove 4.Exit
Choice: 2
Books: []

1.Add 2.View 3.Remove 4.Exit
Choice: 1
Book name: bob
Added.

1.Add 2.View 3.Remove 4.Exit
Choice: 2
Books: ['bob']
```

```
1.Add 2.View 3.Remove 4.Exit
Choice: 3
Book to remove: bob
Removed.

1.Add 2.View 3.Remove 4.Exit
Choice: 4
```

Week - 9

1. Write a Python project to build a Calculator using modular programming (separate module for operations).

Algorithm:

Start

Create separate module for operations

Import module

Call required function

Display result

Stop

operations.py

```
def add(a,b): return a+b
def sub(a,b): return a-b
def mul(a,b): return a*b
def div(a,b): return a/b
```

main.py

```
def add(a,b): return a+b
def sub(a,b): return a-b
def mul(a,b): return a*b
def div(a,b): return a/b

a = float(input("A: "))
b = float(input("B: "))
print("Sum =", add(a,b))

A: 8
B: 6
Sum = 14.0
```

Week - 10

1. Write a Python program that applies modular programming principles and defines multiple reusable functions.

lgorithm:

Start

Define multiple functions

Call functions as required

Display output

Stop

```
def square(x): return x*x
def cube(x): return x*x*x
def avg(a,b): return (a+b)/2

print(square(5), cube(3), avg(10,20))
```

```
25 27 15.0
```

Week - 11

1. Write a Python program using modular programming principles and demonstrate: • Input validation • Testing (minimum 3 test cases) • Debugging practice with comments

Algorithm:

Start

Validate input

Perform operation

Test with multiple inputs

Debug using comments

Stop

```
def safe_int(msg):
    while True:
        try:
            return int(input(msg))
        except:
            print("Invalid input, try again.")

def add(a,b): return a+b

# Debugging comment: Testing add()
# 1) add(2,3) = 5
```

```
# 2) add(-1,1) = 0
# 3) add(10,20) = 30

a = safe_int("Enter A: ")
b = safe_int("Enter B: ")
print("Sum =", add(a,b))
```

```
Enter A: -8
Enter B: 7
Sum = -1
```

Week - 12

1. Write a Python project for a User Registration System with input validation, testing, and debugging documentation.

Algorithm:

Start

Display menu

Register user with validation

Login user

Repeat until exit

Stop

```
users = {}

def register():
    username = input("Username: ")
    if username in users:
        print("Already exists")
        return
    pwd = input("Password: ")
    if len(pwd) < 4:
        print("Weak password")
        return
    users[username] = pwd
    print("Registered")

def login():
    username = input("Username: ")
    pwd = input("Password: ")
    if users.get(username) == pwd:
        print("Login success")
    else:
        print("Login failed")
```

```
while True:
    print("1.Register 2.Login 3.Exit")
    ch = input("Choice: ")
    if ch == "1": register()
    elif ch == "2": login()
    else: break
```

```
1.Register 2.Login 3.Exit
Choice: 1
Username: bishwas
Password: 777777777
Registered
1.Register 2.Login 3.Exit
Choice: 2
Username: bishwas
Password: 77777777
Login failed
1.Register 2.Login 3.Exit
Choice: 2
Username: bishwas
Password: 777777777
Login success
1.Register 2.Login 3.Exit
Choice: 3
```

27.write a mini project in python incorporation various programming concepts (loops,function. lists, modules, validation, testing)

```
import random
import string

# -------------------- DATA MODELS --------------------

class Medicine:
    def __init__(self, name, dosage, frequency, duration,
instructions):
        self.name = name
        self.dosage = dosage
        self.frequency = frequency
        self.duration = duration
        self.instructions = instructions


class Appointment:
    def __init__(self, department, doctor, date, time):
        self.id = ''.join(random.choices(string.ascii_lowercase +
string.digits, k=8))
        self.department = department
        self.doctor = doctor
        self.date = date
```

```
        self.time = time
        self.token = f"T{random.randint(100,999)}"
        self.medicines = self.sample_medicines()

    def sample_medicines(self):
        return [
            Medicine("Amoxicillin", "500mg", "3 times daily", "7
days", "After meals"),
            Medicine("Paracetamol", "650mg", "Every 6 hours", "5
days", "If fever"),
            Medicine("Vitamin D3", "60,000 IU", "Once weekly", "8
weeks", "With milk")
        ]


# -------------------- APPLICATION --------------------

class HospitalApp:
    def __init__(self):
        self.user = {}
        self.appointments = []
        self.departments = {
            "1": "Cardiology",
            "2": "Neurology",
            "3": "Orthopedics",
            "4": "General Medicine"
        }
        self.doctors = {
            "Cardiology": ["Dr. Rao", "Dr. Mehta"],
            "Neurology": ["Dr. Sharma", "Dr. Gupta"],
            "Orthopedics": ["Dr. Kumar", "Dr. Singh"],
            "General Medicine": ["Dr. Das", "Dr. Iyer"]
        }

    # ---------------- LOGIN ----------------
    def login(self):
        print("\n--- Login ---")
        self.user["name"] = input("Enter Name: ")
        self.user["phone"] = input("Enter Phone: ")
        print(f"\nWelcome {self.user['name']}!")

    # ---------------- DEPARTMENTS ----------------
    def select_department(self):
        print("\n--- Departments ---")
        for k, v in self.departments.items():
            print(f"{k}. {v}")
        choice = input("Select Department: ")
        return self.departments.get(choice)

    # ---------------- DOCTORS ----------------
```

```
    def select_doctor(self, department):
        print(f"\n--- Doctors in {department} ---")
        doctors = self.doctors[department]
        for i, d in enumerate(doctors, 1):
            print(f"{i}. {d}")
        return doctors[int(input("Select Doctor: ")) - 1]

    # ---------------- BOOK APPOINTMENT ----------------
    def book_appointment(self):
        dept = self.select_department()
        doctor = self.select_doctor(dept)
        date = input("Enter Date (DD-MM-YYYY): ")
        time = input("Enter Time (HH:MM): ")

        appointment = Appointment(dept, doctor, date, time)
        self.appointments.append(appointment)

        print("\n Appointment Booked Successfully!")
        print(f"Token Number: {appointment.token}")

    # ---------------- HISTORY ----------------
    def view_history(self):
        print("\n--- Appointment History ---")
        if not self.appointments:
            print("No appointments found.")
            return

        for i, a in enumerate(self.appointments, 1):
            print(f"""
{i}. Token: {a.token}
   Department: {a.department}
   Doctor: {a.doctor}
   Date: {a.date}
   Time: {a.time}
""")

    # ---------------- MEDICINES ----------------
    def view_medicines(self):
        print("\n--- Prescribed Medicines ---")
        for a in self.appointments:
            print(f"\nToken: {a.token} | Doctor: {a.doctor}")
            for m in a.medicines:
                print(f"- {m.name} ({m.dosage}), {m.frequency},
{m.duration}")
                print(f"  Instructions: {m.instructions}")

    # ---------------- CANCEL ----------------
    def cancel_appointment(self):
        self.view_history()
        token = input("Enter Token Number to Cancel: ")
```

```
        self.appointments = [a for a in self.appointments if a.token !
= token]
        print("✅ Appointment Cancelled Successfully!")

    # ---------------- MENU ----------------
    def menu(self):
        while True:
            print("""
==============================
1. Book Appointment
2. View Appointment History
3. View Medicines
4. Cancel Appointment
5. Logout
==============================
""")
            choice = input("Select Option: ")

            if choice == "1":
                self.book_appointment()
            elif choice == "2":
                self.view_history()
            elif choice == "3":
                self.view_medicines()
            elif choice == "4":
                self.cancel_appointment()
            elif choice == "5":
                print("Logged out successfully!")
                break
            else:
                print("Invalid choice!")


# -------------------- RUN APP --------------------

if __name__ == "__main__":
    app = HospitalApp()
    app.login()
    app.menu()
```

THE END