LOGIC BUILDING PROGRAMS

Week - 1

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

```
1. Start
2. Input a number n
3. Compute the remainder when n is divided by 2 → r = n % 2
4. If r == 0
     → Print "Even number"

    Else

     → Print "Odd number"
5. End
```

In [ ]:

```
# Program to check if a number is even or odd

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

if num % 2 == 0:
    print(num, "is even.")
else:
    print(num, "is odd.")
```

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

```
1. Start

2. Input a number n

3. If n > 0
  → Print "Positive number"

4. Else if n < 0
  → Print "Negative number"

5. Else
  → Print "Zero"

6. End
```

In [5]:

```
# Program to check if a number is positive, negative, or zero

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

if num > 0:
    print("The number is positive.")
elif num < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

```
Enter a number:  2

The number is positive.
```

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

```
1. Start
```

```
2. Input three numbers: a, b, and c
3. If a ≥ b and a ≥ c

 → Print a is the largest

4. Else if b ≥ a and b ≥ c

 → Print b is the largest

5. Else
  → Print c is the largest
6. End
```

In [6]:
```python
# Program to find the largest of three numbers

a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
c = float(input("Enter third number: "))

if a >= b and a >= c:
    print(a, "is the largest number.")
elif b >= a and b >= c:
    print(b, "is the largest number.")
else:
    print(c, "is the largest number.")
```

```
Enter first number:  2
Enter second number:  2
Enter third number:  3

3.0 is the largest number.
```

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

```
1. Start
2. Input a number n

3. If n ≤ 1 → Not a prime, go to Step 8

4. Set a variable flag = 0

5. Loop i from 2 to √n

     If n % i == 0
      → Set flag = 1
       → Break the loop

6.  If flag == 0
    → n is a prime number

7. Else
  → n is not a prime number

8. Stop
```

In [8]:
```python
num = int(input("Enter a number: "))

if num <= 1:
    print(num, "is not a prime number")
else:
    flag = False

    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            flag = True
            break

    if flag == False:
        print(num, "is a prime number")
    else:
        print(num, "is not a prime number")
```

```
Enter a number:  2
```

```
2 is a prime number
```

Week - 2

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

```
1. Start
2. Input a number n

3. If n < 0
    → Print "Factorial does not exist for negative numbers"
     → Go to Step 8

4. If n == 0
   → Factorial = 1
   → Print factorial
    → Go to Step 8

5. Set factorial = 1

6. For i from 1 to n
   → factorial = factorial * i

7. Print factorial

8. Stop
```

In [9]:

```python
num = int(input("Enter a number: "))

factorial = 1

if num < 0:
    print("Factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    for i in range(1, num + 1):
        factorial *= i
    print("The factorial of", num, "is", factorial)
```

```
Enter a number:  4

The factorial of 4 is 24
```

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

```
1. Start

2. Input a number n

3. Store the number in a temporary variable temp = n

4. Initialize rev = 0

5. Repeat while n > 0
   a. digit = n % 10
   b. rev = rev * 10 + digit
   c. n = n // 10

6. If rev == temp
    → The number is a palindrome
    Else
    → The number is not a palindrome

7. Stop
```

In

```python
num = int(input("Enter a number: "))
```

[10]:

```
temp = num
rev = 0

while num > 0:
    digit = num % 10
    rev = rev * 10 + digit
    num = num // 10

if temp == rev:
    print(temp, "is a palindrome number")
else:
    print(temp, "is not a palindrome number")
```

```
Enter a number:  9

9 is a palindrome number
```

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

```
1. Start

2. Input a string s

3. Convert the string to lowercase (optional but recommended)

4. Reverse the string

    rev = s[::-1]

5. If s == rev
   → The string is a palindrome
   Else
    → The string is not a palindrome

6. Stop
```

In [11]:

```
string = input("Enter a string: ")

# Convert to lowercase to ignore case differences
s = string.lower()

# Reverse the string
rev = s[::-1]

if s == rev:
    print(string, "is a palindrome string")
else:
    print(string, "is not a palindrome string")
```

```
Enter a string:  2

2 is a palindrome string
```

Week - 3

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

```
1. Start

2. Input the number of terms N

3. If N is less than or equal to 0
   → Print “Invalid Input”

4. If N is 1
```

```
→ Print first term: 0

5. Otherwise:

    Initialize:
    a = 0 (first term)
    b = 1 (second term)

     Print a and b

6. Use a loop from 3 to N:

    Compute next term: c = a + b

     Print c

      Update values:
       a = b
       b = c

7. End
```

In [2]:

```
# Fibonacci series up to N terms

N = int(input("Enter the number of terms: "))

if N <= 0:
    print("Invalid input! Enter a positive number.")
elif N == 1:
    print("Fibonacci series:")
    print(0)
else:
    print("Fibonacci series:")
    a, b = 0, 1
    print(a, b, end=" ")

    for _ in range(3, N + 1):
        c = a + b
        print(c, end=" ")
        a, b = b, c
```

```
Enter the number of terms:  3

Fibonacci series:
0 1 1
```

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

```
1. Start
2. Input a number n

3. Initialize sum = 0

4. Repeat while n > 0:

    Extract last digit: digit = n % 10

     Add digit to sum: sum = sum + digit

     Remove last digit: n = n // 10

5. After the loop ends, sum contains the total of all digits

6. Print the sum

7. End
```

In [3]:

```
# Program to find the sum of digits of a number

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

sum_of_digits = 0
```

```
temp = n

while temp > 0:
    digit = temp % 10
    sum_of_digits += digit
    temp //= 10

print("Sum of digits:", sum_of_digits)
```

```
Enter a number:  4

Sum of digits: 4
```

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

```
1. Start
2. Input a string s

3. Convert the string to lowercase (optional, for easy checking)

4. Initialize:

   vowels = 0

   consonants = 0

5. For each character ch in the string:

    If ch is an alphabet:

    If ch is in "aeiou" → increment vowels

    Else → increment consonants

6. Print vowel count and consonant count

7. End
```

In [6]:
```
# Program to count vowels and consonants in a string

s = input("Enter a string: ")

vowels = 0
consonants = 0

for ch in s.lower():
    if ch.isalpha():  # Check if character is a letter
        if ch in "aeiou":
            vowels += 1
        else:
            consonants += 1

print("Number of vowels:", vowels)
print("Number of consonants:", consonants)
```

```
Enter a string:  6

Number of vowels: 0
Number of consonants: 0
```

Week - 4

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

```
1. Start

2. Input a string s
```

```
3. Initialize an empty string rev = ""

4. Find the length of the string using a loop (optional)
   (or directly use len(s) since it’s not a reversing function)

5. Loop from the last index to the first index:

   For i from len(s) - 1 down to 0:

   Append s[i] to rev

6. After the loop, rev contains the reversed string

7. Print the reversed string

8. End
```

In [7]:
```python
# Program to reverse a string without using built-in reverse functions

s = input("Enter a string: ")

rev = ""

# Loop from last character to first
for i in range(len(s) - 1, -1, -1):
    rev += s[i]

print("Reversed string:", rev)
```

```
Enter a string:  2

Reversed string: 2
```

12. Write a Python program to count the occurrence of each character in a string. and algorithm

```
1. Start

2. Input a string s

3. Initialize an empty dictionary count = {}

4. For each character ch in the string:

    If ch already exists in the dictionary
     → increment count[ch] by 1

     Else
       → set count[ch] = 1

5. After processing all characters, the dictionary contains character counts

6. Print each character and its count

7. End
```

In [9]:
```python
# Program to count occurrence of each character in a string

s = input("Enter a string: ")

count = {}

for ch in s:
    if ch in count:
        count[ch] += 1
    else:
        count[ch] = 1

# Display result
for ch in count:
    print(ch, ":", count[ch])
```

```
Enter a string:  7

7 : 1
```

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

```
1. Start
2. Input two numbers: num1 and num2

3. Input an operator (+, -, *, /)

4. Use conditional statements:

    If operator is "+" → compute num1 + num2

    If operator is "-" → compute num1 - num2

     If operator is "*" → compute num1 * num2

     If operator is "/"

    If num2 != 0 → compute num1 / num2

     Else → print “Division by zero not allowed”

  Else → print “Invalid operator”

5. Print the result

6. End
```

In [1]:

```python
# Simple Calculator using conditional statements

print("===== Simple Calculator =====")
print("Select an operation:")
print("1. Addition (+)")
print("2. Subtraction (-)")
print("3. Multiplication (*)")
print("4. Division (/)")

# Taking user input
choice = input("Enter your choice (1/2/3/4): ")

# Taking numbers as input
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

# Conditional statements to perform operations
if choice == '1':
    print("Result:", num1 + num2)

elif choice == '2':
    print("Result:", num1 - num2)

elif choice == '3':
    print("Result:", num1 * num2)

elif choice == '4':
    if num2 != 0:
        print("Result:", num1 / num2)
    else:
        print("Error! Division by zero is not allowed.")

else:
    print("Invalid choice! Please select 1, 2, 3, or 4.")
```

```
===== Simple Calculator =====
Select an operation:
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)

Enter your choice (1/2/3/4):  1
Enter first number:  11
Enter second number:  11

Result: 22.0
```

Week - 5

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

```
1. Start
2. Repeat the following steps until the user chooses to exit:

     1.Display the menu:

         1.Addition

        2.Subtraction

         3.Multiplication

          4.Division

         5.Exit

2. Ask the user to enter a choice

3. If choice is 5 → Exit the loop

4. Else:

     Input two numbers: num1, num2

      If choice is:

         1 → result = num1 + num2

          2 → result = num1 - num2

          3 → result = num1 * num2

           4 →

              If num2 != 0 → result = num1 / num2

              Else → print “Cannot divide by zero”

        Print the result

3. End
```

In [ ]:

```
# Menu-driven calculator using loop

while True:
    print("\n--- Calculator Menu ---")
    print("1. Addition")
    print("2. Subtraction")
    print("3. Multiplication")
    print("4. Division")
    print("5. Exit")

    choice = int(input("Enter your choice: "))

    if choice == 5:
        print("Exiting the program... Goodbye!")
        break

    num1 = float(input("Enter first number: "))
    num2 = float(input("Enter second number: "))

    if choice == 1:
        print("Result:", num1 + num2)

    elif choice == 2:
        print("Result:", num1 - num2)
    elif choice == 3:
        print("Result:", num1 * num2)

    elif choice == 4:
        if num2 != 0:
            print("Result:", num1 / num2)
        else:
            print("Error! Cannot divide by zero.")

    else:
        print("Invalid choice! Please select from 1–5.")
```

```
--- Calculator Menu ---
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit

Enter your choice:  1
Enter first number:  2
Enter second number:  2

Result: 4.0

--- Calculator Menu ---
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
```

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

```
1. Start
2. Repeat the following steps:
   1. Repeat the following steps:
   2. For i from 1 to 10:
        Compute n * i
         Print the result
    3. Ask the user: “Do you want to continue? (yes/no)”
    4. If the user enters no → Exit the loop
3. End
```

In [1]:

```
# Program to generate multiplication table until user stops

while True:
    n = int(input("Enter a number to print its multiplication table: "))

    print(f"\nMultiplication Table of {n}:")
    for i in range(1, 11):
        print(f"{n} x {i} = {n * i}")

    choice = input("\nDo you want to continue? (yes/no): ").lower()

    if choice == "no":
        print("Program ended. Goodbye!")
        break
```

```
Enter a number to print its multiplication table:  18


Multiplication Table of 18:
18 x 1 = 18
18 x 2 = 36
18 x 3 = 54
18 x 4 = 72
18 x 5 = 90
18 x 6 = 108
18 x 7 = 126
18 x 8 = 144
18 x 9 = 162
18 x 10 = 180
```

```
Do you want to continue? (yes/no):  no

Program ended. Goodbye!
```

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

```
1. Start
2. Input the number of rows n
3. For each row i from 1 to n:
   For each column j based on the required pattern:
   Print star/number/space as required
    Move to next line after each row
4. End
```

In [2]:
```
n = int(input("Enter number of rows: "))

for i in range(1, n + 1):
    print("*" * i)
```

```
Enter number of rows:  6

*
**
***
****
*****
******
```

Pattern 1: Left-Aligned Star Triangle

Example Output (n = 5)

```
1. Input n
2. Loop i from 1 to n
3. For each row, print i stars
4. Move to next line
```

In [2]:
```
n = int(input("Enter number of rows: "))

for i in range(1, n + 1):
    print("*" * i)
```

```
Enter number of rows:  5

*
**
***
****
*****
```

Pattern 2: Inverted Star Triangle

Example Output (n = 5)

```
1. Input n

2. Loop i from n down to 1

3. Print i stars on each row
```

```
4. Move to next line
```

In [3]:

```python
n = int(input("Enter number of rows: "))

for i in range(n, 0, -1):
    print("*" * i)
```

```
Enter number of rows:  5

*****
****
***
**
*
```

Pattern 3: Number Triangle

Example Output (n = 5)

```
1. Input n

2. Loop i from 1 to n

3. For each row, print numbers from 1 to i

4. Move to next line
```

In [4]:

```python
n = int(input("Enter number of rows: "))

for i in range(1, n + 1):
    for j in range(1, i + 1):
        print(j, end="")
    print()
```

```
Enter number of rows:  5

1
12
123
1234
12345
```

Pattern 4: Pyramid Star Pattern

Example Output (n = 5)

```
1. Input n

2. Loop i from 1 to n

3. Print (n - i) spaces

4. Print (2*i - 1) stars

5. Move to next line
```

In [5]:

```python
n = int(input("Enter number of rows: "))

for i in range(1, n + 1):
    print(" " * (n - i) + "*" * (2*i - 1))
```

```
Enter number of rows:  5
```

```
    *
   ***
  *****
 *******
*********
```

FUNCTION-BASED QUESTIONS

Week - 6

17. Write a Python function that takes a user’s name and prints a greeting message.

1. Start

2. Define a function greet(name)

3. Inside the function:

Print a message: "Hello, ! Welcome!"

4. Ask the user to input their name

5. Call the function and pass the user’s name as an argument

6. End

In [3]:

```
# Function to greet the user

def greet(name):
    print("Hello,", name + "! Welcome!")

user_name = input("Enter your name: ")
greet(user_name)
```

```
Enter your name:  MINNU

Hello, MINNU! Welcome!
```

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

1.Start

2. Define a function add(a, b)

3. Inside the function:

Compute sum = a + b

Return the sum

4. In the main program:

Accept two numbers from the user

Call the function with these numbers

Print the returned value

5. End

In [2]:

```
# Function to return the sum of two numbers

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

num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

result = add(num1, num2)
print("The sum is:", result)
```

```
Enter first number:  2
Enter second number:  3

The sum is: 5.0
```

Week – 7

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

```
1. Start

2. Define a function factorial(n)

3. Check if n is 0 or 1

   If yes → return 1 (base case)

4. Otherwise (recursive case):

   Return n * factorial(n - 1)

5. In the main program:

   Input a number n

   Call the function factorial(n)

   Print the result

6. End
```

In [3]:

```
# Recursive function to find factorial of a number

def factorial(n):
    if n == 0 or n == 1:     # Base case
        return 1
    else:
        return n * factorial(n - 1)   # Recursive call

num = int(input("Enter a number: "))
print("Factorial of", num, "is:", factorial(num))
```

```
Enter a number:  4

Factorial of 4 is: 24
```

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

1. Start

2.Define a lambda function:
is_even = lambda n: n % 2 == 0

3. Input a number n from the user

4. Call the lambda function with the number

5. If it returns True → number is even
If it returns False → number is odd

6. Print the result

7. End

In [4]:

```python
# Lambda function to check if a number is even

is_even = lambda n: n % 2 == 0

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

if is_even(num):
    print(num, "is even.")
else:
    print(num, "is odd.")
```

```
Enter a number:  4

4 is even.
```

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

```
1. Start

2. Define a recursive function factorial(n)

    If n == 0 or n == 1 → return 1

    Otherwise → return n * factorial(n - 1)

3. In the main program:

  1. Ask the user to enter a number

   2.Validate input:

     If the number is negative → print “Factorial not
      possible for negative numbers”

      If the number is not an integer → show an error

3. If the number is valid:

   Call the function factorial(n)

   Print the result

4. End
```

In [5]:

```python
# Recursive Factorial Program with Input Validation

def factorial(n):
    if n == 0 or n == 1:        # Base case
        return 1
    else:
        return n * factorial(n - 1)   # Recursive case

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

    if num < 0:
        print("Error: Factorial is not defined for negative numbers.")
    else:
        print("Factorial of", num, "is:", factorial(num))

except ValueError:
    print("Invalid input! Please enter a valid integer.")
```

```
Enter a number:  5
```

```
Factorial of 5 is: 120
```

PROJECT / ADVANCED QUESTIONS

Week-8

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

```
1. Start

2. Create an empty list called library to store book details

3. Define a function add_book()

     Input Book ID, Title, Author

      Store details in a dictionary

      Append dictionary to the library list

4. Define a function display_books()

      If the library is empty → print “No books available”

      Else → display all book records

5. Define a function search_book()

      Input Book ID to search

      Loop through the library list

       If Book ID matches → display book details

       If not found → print “Book not found”

6. Define a function remove_book()

     Input Book ID to remove

     Loop through the library list

     If Book ID matches → remove the book and print success message

     Else → print “Book not found”

7. Display a menu with options:

   Add Book

   Display All Books

   Search Book

   Remove Book

   Exit

8. Input user choice

9. Call the corresponding function based on the user’s choice

10. Repeat menu until the user chooses to exit
```

```
11. Stop
```

In [2]:

```
# Library Book Management System using Functions

library = []  # List to store book records

def add_book():
    book_id = input("Enter Book ID: ")
    title = input("Enter Book Title: ")
    author = input("Enter Author Name: ")
    library.append({"Book ID": book_id, "Title": title, "Author": author})
    print("Book added successfully!\n")

def display_books():
    if not library:
        print("No books available in the library.\n")
        return

    print("\n--- Library Books ---")
    for book in library:
        print(f"Book ID: {book['Book ID']}, Title: {book['Title']}, Author: {book['Author']}")
    print()

def search_book():
    book_id = input("Enter Book ID to search: ")
    for book in library:
        if book["Book ID"] == book_id:
            print("Book Found:")
            print(f"Book ID: {book['Book ID']}, Title: {book['Title']}, Author:
{book['Author']}\n")
            return
    print("Book not found!\n")

def remove_book():
    book_id = input("Enter Book ID to remove: ")
    for book in library:
        if book["Book ID"] == book_id:
            library.remove(book)
            print("Book removed successfully!\n")
            return
    print("Book not found!\n")


# Main Program
while True:
    print("===== Library Book Management System =====")
    print("1. Add Book")
    print("2. Display All Books")
    print("3. Search Book")
    print("4. Remove Book")
    print("5. Exit")

    choice = input("Enter your choice: ")

    if choice == "1":
        add_book()
    elif choice == "2":
        display_books()
    elif choice == "3":
        search_book()
    elif choice == "4":
        remove_book()
    elif choice == "5":
        print("Exiting the program... Goodbye!")
        break
    else:
        print("Invalid choice! Please try again.\n")
```

```
===== Library Book Management System =====
1. Add Book
2. Display All Books
3. Search Book
4. Remove Book
5. Exit

Enter your choice:  1
Enter Book ID:  25rbaim0060
Enter Book Title:  Can We Be Strangers again
Enter Author Name:  HASSINI

Book added successfully!

===== Library Book Management System =====
1. Add Book
2. Display All Books
3. Search Book
4. Remove Book
5. Exit

Enter your choice:  5

Exiting the program... Goodbye!
```

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

```
1. Start

2. Create a separate file operations.py

3. Inside operations.py, define functions:

   add(a, b) → return a + b

   subtract(a, b) → return a – b

   multiply(a, b) → return a × b

   divide(a, b)

     If b = 0 → return error

      Else → return a ÷ b

4. Create main file calculator.py

5. Import the operations module using import operations

6. Display a menu with options:

   1 → Addition

   2 → Subtraction

   3 → Multiplication

   4 → Division

   5 → Exit

7. Take user input for choice

8. If choice = 5 → Stop the program

9. Else

   Accept two numbers from the user

   Call the respective function from operations module

    Print the result

10. Repeat steps 6–9 until the user chooses Exit

11. Stop
```

In [26]:

```
# operations.py

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

def subtract(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    if b == 0:
        return "Error: Division by zero!"
    return a / b
```

In [ ]:

```
import os
import pytest


def calculator():
    print("welcomne to mini calculator!\n")

    while True:

        print("\nchoose operation:")
        print("1: Add")
        print("2: Subtract")
        print("3: Multiply")
        print("4: Divide")
        print("5: Exit")

        choice = input("Enter choice (1/2/3/4/5): ")

        if choice == "5":
            print("Exiting calculator...\n")
            break
        try:
            a = float(input("Enter first number:"))
            b = float(input("Enter second number:"))
        except valueError:
            print("please enter vaild numbers!\n")
            continue
        if choice == "1":
            print(f"Result: {add(a, b)}\n")
        elif choice == "2":
            print(f"Result: {subtract(a, b)}\n")
        elif choice == "3":
            print(f"Result: {multiply(a, b)}\n")
        elif choice == "4":
            try:
                print(f"Result: {divide(a, b)}\n")
            except ValueError as e:
                print(f"Error: {e}\n")

        else:
            print("Invalid choice please try again.\n")


def run_tests():
    print("Running automated tests...")

    # Absolute path to tests_operations.py
    project_root = os.path.abspath(os.path.join(os.path.dirname(__file__),"..",".."))
    test_file_path = os.path.join(project_root,"tests", "test_operations.py")

    # Run pytest programmatically
    result = pytest.main([test_file_path, "-q","--tb=short"])
    if result == 0:
       print("All tests passed! ✅")
    else:
       print("Some tests failed! ❌")


if __name__ == "__main__":
     calculator()
     run_tests()
```

```
welcomne to mini calculator!


choose operation:
1: Add
2: Subtract
```

```
3: Multiply
4: Divide
5: Exit

Enter choice (1/2/3/4/5):  1
Enter first number: 2
Enter second number: 2

Result: 4.0


choose operation:
1: Add
2: Subtract
3: Multiply
4: Divide
5: Exit
```

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

```
1. Start

2. Display a menu:

    1. Register User

     2. View Users

      3. Exit

3. Accept the user’s choice.

4. If choice = 1 (Register User):

    Ask for name.

     Validate: must contain only alphabets and spaces.

     Ask for email.

     Validate: must contain "@", ".", and no spaces.

      Ask for password.

      Validate:

           Minimum 6 characters

            Must contain at least one digit

            Must contain at least one uppercase letter

      If all inputs valid → Save user details to list.

       Else → Show error and retry.

5. If choice = 2 (View Users):

     Display all registered users.

6. If choice = 3 (Exit):

   Stop program.

7. End.
```

In [3]:

```
# ------------------------------
# User Registration System Project
# ------------------------------

registered_users = []   # Stores all registered users


# ------------------------------
```

```python
# Input Validation Functions
# ------------------------------

def validate_name(name):
    return name.replace(" ", "").isalpha()


def validate_email(email):
    return "@" in email and "." in email and " " not in email


def validate_password(password):
    if len(password) < 6:
        return False
    if not any(ch.isdigit() for ch in password):
        return False
    if not any(ch.isupper() for ch in password):
        return False
    return True


# ------------------------------
# Function to Register User
# ------------------------------

def register_user():
    print("\n--- Register New User ---")

    name = input("Enter your name: ")
    if not validate_name(name):
        print("❌ Invalid name! Only alphabets allowed.")
        return

    email = input("Enter your email: ")
    if not validate_email(email):
        print("❌ Invalid email format!")
        return

    password = input("Enter a password: ")
    if not validate_password(password):
        print("❌ Weak password! Must be 6+ chars, contain an uppercase letter and a digit.")
        return

    registered_users.append({
        "name": name,
        "email": email,
        "password": password
    })

    print("✅ User registered successfully!")


# ------------------------------
# Function to View Registered Users
# ------------------------------

def view_users():
    print("\n--- Registered Users ---")
    if not registered_users:
        print("No users registered yet.")
        return

    for i, user in enumerate(registered_users, 1):
        print(f"{i}. {user['name']} - {user['email']}")


# ------------------------------
# Main Program Loop
# ------------------------------

def main():
    while True:
        print("\n===== USER REGISTRATION SYSTEM =====")
        print("1. Register User")
        print("2. View Registered Users")
        print("3. Exit")

        choice = input("Enter your choice: ")
        if choice == "1":
            register_user()

        elif choice == "2":
            view_users()

        elif choice == "3":
            print("Exiting program...")
            break

        else:
            print("❌ Invalid option! Please choose 1, 2, or 3.")


# Run the program
main()
```

```
===== USER REGISTRATION SYSTEM =====
1. Register User
2. View Registered Users
3. Exit

Enter your choice:  1


--- Register New User ---

Enter your name:  Hasini
Enter your email:  25rbaim0060@mnru.ac.in
Enter a password:  HASINI@18

✅ User registered successfully!

===== USER REGISTRATION SYSTEM =====
1. Register User
2. View Registered Users
3. Exit

Enter your choice:  3

Exiting program...
```

Week - 13

27. Write a mini-project in Python incorporating various programming concepts (loops, functions, lists, modules, validation, testing).

```
1. Start the program.

2. Display menu options:

    Add student

    Add marks

     Calculate average

     Display all student details

     Search student

     Exit

3. Ask user to enter a choice (validate input).

4. Based on the choice, call the appropriate function from
    grade_module.py.
5. Repeat the menu using a loop until the user selects Exit.
```

```
6. End program.
```

In [4]:

```python
# ============================
# To-Do List Management System
# ============================

# List to store tasks (each task is a dictionary)
tasks = []


def add_task():
    """Add a new task to the to-do list."""
    task_name = input("Enter task name: ").strip()

    if task_name == "":
        print("Task name cannot be empty!")
        return

    task = {
        "name": task_name,
        "completed": False
    }

    tasks.append(task)
    print("Task added successfully!")


def view_tasks():
    """Display all tasks in the list."""
    if not tasks:
        print("No tasks available.")
        return

    print("\n----- To-Do List -----")
    for index, task in enumerate(tasks, start=1):
        status = "✔ Completed" if task["completed"] else "✘ Not Completed"
        print(f"{index}. {task['name']} - {status}")

    print("----------------------")


def mark_completed():
    """Mark a task as completed."""
    if not tasks:
        print("No tasks to update!")
        return

    try:
        task_no = int(input("Enter task number to mark completed: "))
        if 1 <= task_no <= len(tasks):
            tasks[task_no - 1]["completed"] = True
            print("Task marked as completed!")
        else:
            print("Invalid task number!")
    except ValueError:
        print("Please enter a valid number!")


def delete_task():
    """Delete a task from the list."""
    if not tasks:
        print("No tasks to delete!")
        return

    try:
        task_no = int(input("Enter task number to delete: "))
        if 1 <= task_no <= len(tasks):
            tasks.pop(task_no - 1)
            print("Task deleted successfully!")
        else:
            print("Invalid task number!")
    except ValueError:
        print("Enter a valid number!")


def show_menu():
    """Display menu options."""
    print("\n===== TO-DO LIST MENU =====")
    print("1. Add Task")
    print("2. View Tasks")
    print("3. Mark Task Completed")
    print("4. Delete Task")
    print("5. Exit")
    print("===========================")


# -------- MAIN PROGRAM LOOP --------
while True:
    show_menu()

    choice = input("Enter your choice (1-5): ").strip()

    if choice == "1":
        add_task()

    elif choice == "2":
        view_tasks()

    elif choice == "3":
        mark_completed()

    elif choice == "4":
        delete_task()

    elif choice == "5":
        print("Exiting program... Goodbye!")
        break

    else:
        print("Invalid choice! Please enter a number between 1 and 5.")
```

```
===== TO-DO LIST MENU =====
1. Add Task
2. View Tasks
3. Mark Task Completed
4. Delete Task
5. Exit
===========================

Enter your choice (1-5):  1
Enter task name:  Red Red Roses

Task added successfully!

===== TO-DO LIST MENU =====
1. Add Task
2. View Tasks
3. Mark Task Completed
4. Delete Task
5. Exit
===========================

Enter your choice (1-5):  5

Exiting program... Goodbye!
```

In [ ]: