logic building programs

WEEK-1

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

Algorithm:

1. start
2. give input two numbers
3. if num%2==0,then give num is even
4. else give num is odd
5. stop

In [16]:
```
num = int(input("enter a num: "))
if num%2 == 0:
    print("num is even")
else
    print("num is odd")
```

```
num is odd
```

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

Algorithm

1. Start
2. Input a num
3. If num>0
4. Display "Positive number"
5. Else if num<0
6. Display "Negative number"
7. Else
8. Display "Zero" 9)Stop

In [18]:
```
num = int(input("enter a num :"))
if num>0:
    print("num is positive")
elif num<0:
    print("num is negative")
else:
    print("num is equal to zero")
```

```
num is negative
```

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

Algorithm

1. Start
2. Input three numbers: A, B, C
3. If A>B and A>C
4. Display "A is the largest"
5. Else if B>A and B>C
6. Display "B is the largest"
7. Else
8. Display "C is the largest"
9. Stop

In [28]:
```
num1 = int(input("enter a num: "))
num2 = int(input("enter a num: "))
num1 = int(input("enter a num: "))
if num1>num2 and num1>num3:
    print("num1 is larger")
elif num2>num3 and num2>num1:
    print("num2 is larger")
else:
    print("num3 is larger")
```

```
num2 is larger
```

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

Algorithm

1. Start
2. Input a number
3. If num\leq 1
4. Display "Not a prime number" and Stop
5. Set a counter variable i=2
6. Repeat while i\leq num/2: 7)If Display "Not a prime number" and Stop

    Increment i by 1
7. If no divisor is found Display "Prime number"
8. Stop

```
In [26]: num = int(input("enter a num :"))
         if num>1:
            for i  in range(2,num):
                if  num%i==0:
                    print("num is not a prime")
                break
            else:
              print("num is a prime")
         else:
             print("num is not a prime")
```

```
num is not a prime
```

WEEK-2

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

Algorithm

1. Start
2. Input a number
3. Import the math module
4. Compute the factorial using math.factorial(N)
5. Display the factorial value
6. Stop

```
In [49]: num = int(input("enter a num: "))
         import math
         print(math.factorial(num))
```

```
120
```

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

Algorithm

1. start

2. input a stirng of length n
3. Set left = 0.
4. Set right = length(S) − 1.
5. Repeat while left < right:
6. If S[left] ≠ S[right]
7. Print “Not a palindrome”,stop
8. Else Increment left by 1 and → Decrement right by 1
9. Print “Palindrome”.
10. Stop

In [54]:
```
num = int(input("enter a num : "))
if num_str==num_str[::-1]:
    print("num is a polyndrome")
else:
    print("num is not a polyndrome")
```

```
num is a polyndrome
```

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

Algorithm

1. Start
2. Input the string s
3. if num_str and num_str[::-1]
4. print "num is a palindrome
5. Else: "Print num is not a palindrome”
6. Stop

In [61]:
```
str = input("enter a string: ")
if str == str[::-1]:
    print("str is a polyndrome")
else:
    print("str is not polyndrome")
```

```
str is a polyndrome
```

WEEK-3

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

Algorithm

1. Start
2. Input"enter a num"
3. Intialize a=0 ,b=1
4. print the value of a
5. compute the next fibinoacci number next = a+b
6. next=a+b
7. a=b,b=next
8. Stop

In [47]:
```
num= int(input("enter a num : "))
a,b = 0,1
for i in range(num):
    print(a,end="")
```

```
0000000
```

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

Algorithm

1. Start
2. give input enter a num
3. if num>0,then digit = num%10
4. total+= digit,num=num//10
5. give output
6. Stop

In [5]:
```
num= int(input("enter a num : "))
total= 0
while num>0:
    digit = num%10
    total+=digit
    num = num//10
print("sum of digits:",total)
```

```
sum of digits: 10
```

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

Algorithm

1. Start
2. Input a string
3. Vowel="aeiouAEIOU"
4. Vowel_count=0 and consonant_count=0
5. If ch is an alphabet character (isalpha()): If ch is in the vowel then vowel_count + 1
6. else count_count+=1
7. Stop

In [10]:
```python
str=input("enter a str:")

vowels="aeiouAEIOU"
vowel_count = 0
consonants_count = 0
for ch in string:
    if ch.isalpha():
        if  ch in vowels:
            vowel_count += 1
        else:
           consonant_count += 1

print("vowel:",vowel_count)
print("consonant:",consonant_count)
```

```
vowel: 4
consonant: 10
```

WEEK-4

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

Algorithm

1. Start
2. Input the string text
3. Initialize an empty string reversed_text ← ""

4. For each character char in text (from left to right):
5. Set reversed_text ← char + reversed_text
6. End For
7. Output reversed_text 8)Stop

In [11]:
```python
text= input("enter a string:")
reversed_text =""
for char in text:
    reversed_text = char+ reversed_text
print("reversed string :",reversed_text)
```

```
reversed string : iah
```

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

Algorithm

1. Start
2. Read the input string from the user.
3. Create an empty dictionary to store character counts.
4. Repeat for each character in the string:
5. If the character is already in the dictionary, increase its count by 1.
6. Otherwise, add the character to the dictionary with count 1.
7. Display the dictionary containing each character and its count.
8. Stop

In [10]:
```python
string = input("enter a string: ")
count = {}
for  ch in string:
     count[ch] = count.get(ch,0) + 1
print(count)
```

```
{'h': 1, 'a': 2, 'r': 2, 'i': 2, 'p': 1, 'y': 1}
```

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

Algorithm

1. Start
2. Give input as num1 and num2 from the user
3. Read the operator (+, -, *, /) from the user.

4. If the operator is "+" → Compute sum = num1 + num2 → Display the result.
5. Else if the operator is "-" → Compute difference = num1 - num2 → Display the result.
6. Else if the operator is "*" → Compute product = num1 * num2 → Display the result.
7. Else if the operator is "/" If num2 ≠ 0 → Compute quotient = num1 / num2 → Display the result.
8. Else → Display "Error: Cannot divide by zero".
9. Else → Display "Invalid operator".
10. Stop

In [16]:
```python
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

print("Select operation:add,sub,multiply,division")
op = input("Enter operator: ")

if op == "add":
    print("Result:", num1 + num2)
elif op == "sub":
    print("Result:", num1 - num2)
elif op == "multiply":
    print("Result:", num1 * num2)
elif op == "devison":
    if num2 != 0:
        print("Result:", num1 / num2)
    else:
        print("Error: Cannot divide by zero")
else:
    print("Invalid operator")
```

```
Select operation:add,sub,multiply,division
Result: 1462.0
```

week-4

14. Write a Python program to implement a menu-driven calculator using a loop

(repeat until the user exits).

Algorithm

1. Start

2. Display the calculator menu: 1. Addition 2. Subtraction 3. Multiplication 4. Division 5. Exit
3. Read the user’s choice. If the choice is 5, → Display “Exiting…” → Break the loop.
4. Read the first number.
5. Read the second number.
6. If the choice is 1, → Compute sum = num1 + num2 → Display the result.
7. Else if the choice is 2, → Compute difference = num1 - num2 → Display the result.
8. Else if the choice is 3, → Compute product = num1 * num2 → Display the result.
9. Else if the choice is 4, If num2 ≠ 0, → Compute quotient = num1 / num2 → Display the result. Else → Display “Cannot divide by zero”.
10. Else, → Display “Invalid choice”.
11. Stop

In [20]:
```
while True:
    print("\n--- Simple Calculator ---")
    print("1. Addition")
    print("2. Subtraction")
    print("3. Multiplication")
    print("4. Division")
    print("5. Exit")

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

    if choice == "5":
        print("Exiting the calculator...")
        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 1–5.")
```

```
--- Simple Calculator ---
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Result: -53.0

--- Simple Calculator ---
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Exiting the calculator...
```

15. Write a Python program to generate a multiplication table for a given number

(loop until the user stops).

Algorithm

1. Start
2. Ask the user to enter a number.
3. for i in range(1,11): print(num , "*", i = num * i)
4. Ask the user if they want to continue (yes/no).
5. If the user enters no, →Exit the loop.
6. Stop

In [49]:
```python
while True:
    num = int(input("enter a num: "))

    for i in range(1,11):
        print(num, "*" , i ,"=", num * i)

    choice = input("do you want to continue? yes/no: ").lower()
    if choice != "yes":
        break
```

```
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
5 * 10 = 50
```

```
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
6 * 10 = 60
```

16. Write a Python program to print different patterns using loop concepts

(e.g., star patterns, number patterns).

Right traingle:

Algorithm

1. Start
2. Ask the user to enter a number and store it in rows
3. Set a loop variable i starting from 1
4. Repeat the following steps while i is less than or equal to rows:
5. Print i stars (*)
6. Increase i by 1
7. Stop

In [52]:
```python
rows= int(input("enter a num : "))
for i in range(1,rows+1):
    print("*" * i)
```

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

left traingle

Algorithm

1. Start
2. Ask the user to enter the number of rows

3. for i = 1 to rows
4. Print (rows - i) spaces, Print i stars (*)
5. Stop

In [66]:
```
rows = int(input("enter a num: "))

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

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

FUNCTION -BASED QUESTIONS

WEEK-6

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

Algorithm

1. Start
2. Define a function greet(name)
3. Inside the function: Print the message: "Hello, " + name + "!"
4. Input the user’s name step-5: Call the function using the input name step-6: Stop

In [73]:
```
def greet(name):
    print("Hello, " + name + "! how are you!")

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

```
Hello, lekhasai! how are you!
```

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

Algorithm

1. Start
2. Define a function add_numbers(a, b)
3. Inside the function: Compute sum = a + b , Return the value of sum
4. Input two numbers from the user
5. Call the function with the two numbers
6. Print the returned result
7. Stop

In [74]:
```
def add_numbers(a, b):
    return a + b

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

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

```
The sum is: 11.0
```

WEEK -7

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

Algorithm

1. Start
2. Define a function factorial(n)
3. Check base case:
4. If n == 0 or n == 1 → return 1
5. Recursive case:
6. Return n * factorial(n - 1)
7. Input a number from the user
8. Call the function and print the result
9. Stop

In [85]:
```
def factorial(n):
    if  n == 0 or num ==1:
        return 1
    else:
        return (n*factorial(n-1))

num = int(input("enter a num : "))

print("factorial of", num,"is",factorial(num))
```

```
factorial of 5 is 120
```

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

Algorithm

1. Start
2. Define a lambda function:
3. is_even = lambda n: n % 2 == 0 4)Input a number → num
4. Call the lambda function with num
5. If the result is True
6. The number is even
7. Else 9)The number is odd
8. End

is_even = lambda n:n%2 == 0 num= int(input("enter a num: ")) print("even num: ",is_even(num))

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

Algorithm

1. Start
2. Repeat until valid input is received
3. Read input as a string → num
4. Check if num contains only digits
5. If yes, convert to integer and continue 6)If no, display error message and ask again
6. If n == 0 or n == 1
7. Return 1
8. Else, Return n * factorial(n - 1)
9. Call the function using the validated number

10. Display the factorial result
11. End

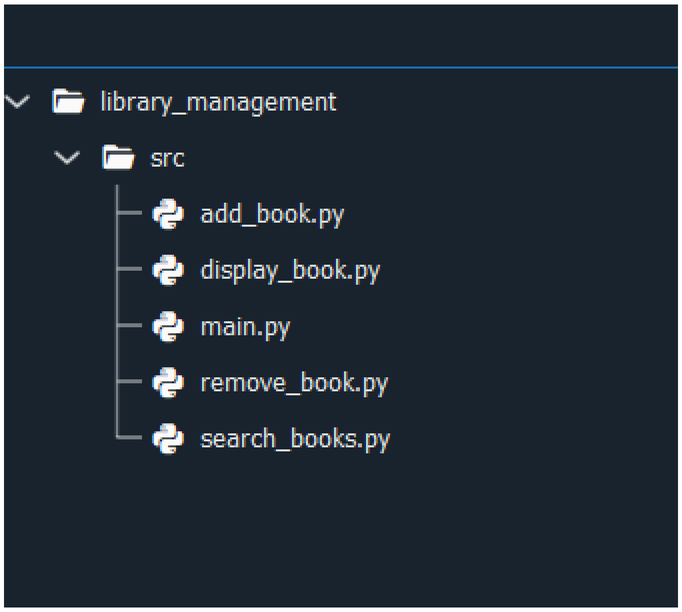
In [96]:
```python
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)

while True:
    num = input("Enter a non-negative integer: ")

    if num.isdigit():
        num = int(num)
        break
    else:
        print("Invalid input! Please enter a non-negative integer.")

print("Factorial of", num, "is", factorial(num))
```

```
Invalid input! Please enter a non-negative integer.
Factorial of 5 is 120
```

PROJECT/ ADVANCED QUESTIONS

WEEK-8

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



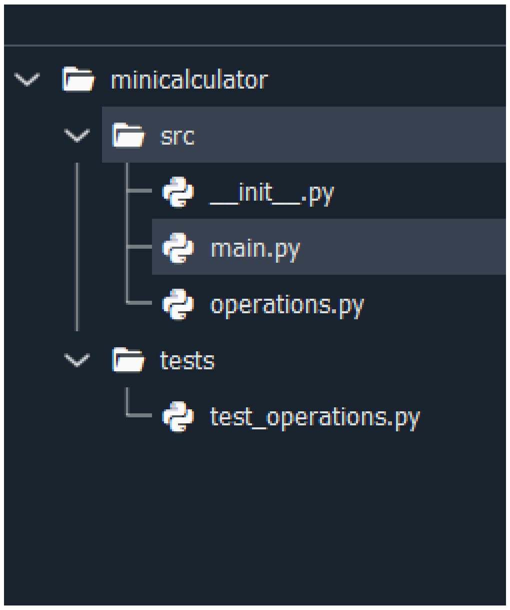
```
library_management
  src
    add_book.py
    display_book.py
    main.py
    remove_book.py
    search_books.py
```

```
In [ ]: library = []
def add_book(title, author, year, isbn):
 for book in library:
 if book['isbn'] == isbn:
 print("Book with this ISBN already exists.")
 return
 library.append({
 'title': title,
 'author': author,
 'year': year,
 'isbn': isbn
 })
 print("Book added successfully.")

def remove_book(isbn):
 global library
 for book in library:
 if book['isbn'] == isbn:
 library.remove(book)
 print("Book removed.")
 return
 print("Book not found.")

def search_books(field, value):
 results = []
 for book in library:
 if str(book.get(field, '')).lower() == str(value).lower():
 results.append(book)
 return results

def display_books():
 if not library:
 print("Library is empty.")
 return
 for idx, book in enumerate(library, 1):
 print(f"{idx}. {book['title']} by {book['author']} ({book['year']}) [ISBN: {b

# Main program loop
def main():
    while True:
    print("\nLibrary System")
    print("1. Add Book")
    print("2. Remove Book")
    print("3. Search Book")
    print("4. Display All Books")
    print("5. Exit")
    choice = input("Enter choice: ")

    if choice == '1':
       title = input("Enter title: ")
       author = input("Enter author: ")
       year = input("Enter year: ")
       isbn = input("Enter ISBN: ")
       add_book(title, author, year, isbn)
    elif choice == '2':
     isbn = input("Enter ISBN to remove: ")
     remove_book(isbn)
    elif choice == '3':
      field = input("Search by (title/author/year): ").lower()
      value = input("Enter search value: ")
      results = search_books(field, value)
    if results:
       for book in results:
          print(f"{book['title']} by {book['author']} ({book['year']}) [ISBN:
    else:
       print("No matching books found.")
    elif choice == '4':
       display_books()
    elif choice == '5':
       print("Exiting...")
       break
    else:
      print("Invalid choice.")

if __name__ == "__main__""
  main()
```

WEEK - 9

23. Write a Python project to build a Calculator using modular programming

(separate module for operations).

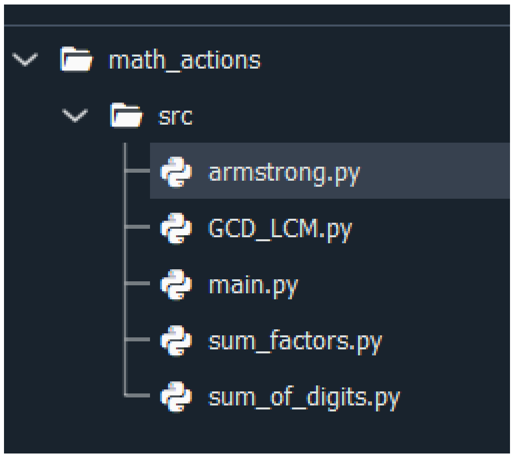


In [ ]:
```
# src/calculator_app/operations.py
```

```
def add(a: float, b: float) -> float:

return a + b

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

def multiply(a: float, b: float) -> float:
    return a * b

def divide(a: float, b: float) -> float:
    if b == 0:
        raise ValueError("Cannot divide by zero.")
    return a /b

# src/calculator_app/main.py
import os
# import pytest
from  operations import add, subtract, multiply, divide
def calculator():
    print("Welcome 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 valid 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 test_operations.py
#     project_root = ("C:\\Users\\ramus\\calculator")
#     test_file_path = ("C:\\Users\\ramus\\Downloads\\calculator\\test_operati
#     # 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()
```

WEEK -10

24. Write a Python program that applies modular programming principles and defines multiple resuable functions



```
math_actions
    src
        armstrong.py
        GCD_LCM.py
        main.py
        sum_factors.py
        sum_of_digits.py
```

In [ ]:
```
def is_armstrong(num):
    digits = str(num)
    n = len(digits)

    total = sum(int(digit) ** n for digit in digits)
```

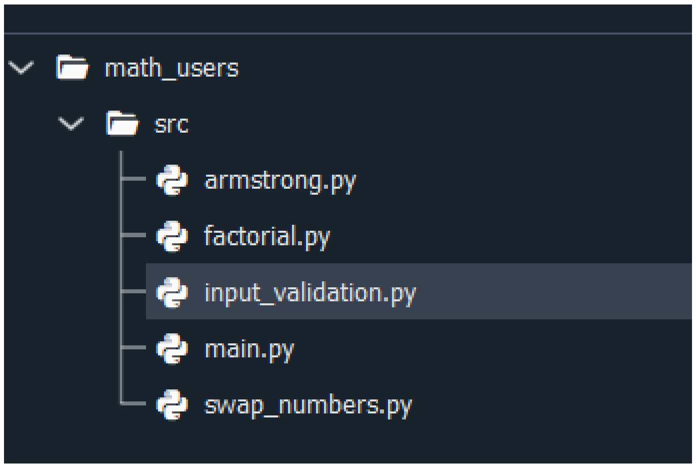
```python
    return total == num

import math

def find_gcd(a, b):
    return math.gcd(a, b)

# Function to find LCM
def find_lcm(a, b):
    return abs(a * b) // math.gcd(a, b)

def is_perfect(n):
    sum_factors = 0
    for i in range(1, n):
        if n % i == 0:
            sum_factors += i
    return sum_factors == n

def sum_of_digits(n):
    total = 0
    while n > 0:
        total += n % 10   # extract last digit
        n //= 10          # remove last digit
    return total

from armstrong import is_armstrong
from sum_of_digits import sum_of_digits
from sum_factors import is_perfect
from GCD_LCM import find_gcd,find_lcm



def main():
    print ("welcome to math_actions\n")

    while 'true':

        print("1.check armstrong number")
        print("2.sum of the digits")
        print("3.check perfect number")
        print("4.find gcd and lcm ")
        print("5.exit")

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


        if choice == "5":
            print("Exiting math_actions...\n")
            break

        if choice == '1':
           num = int(input("enter a num : \n"))
```

```
            if is_armstrong(num):
               print(f"{num} is an armstrong num\n")
            else:
               print(f"{num} is not an armstrong num")
         elif choice == '2':
             num=int(input("enter a num: "))

         print("sum of digits:", sum_of_digits(num))
         elif choice == '3':
              num = int(input("Enter a number: \n"))
              if is_perfect(num):
                 print(num, "is a Perfect Number")
              else:
                 print(num, "is Not a Perfect Number")
         elif  choice =='4':
              num1 = int(input("enter a num : "))
              num2 = int(input("enter a num : "))
              print("GCD of", num1, "and", num2, "is:", find_gcd(num1, num2))
              print("LCM of", num1, "and", num2, "is:", find_lcm(num1, num2))
         else:
             print("Invalid choice. Try again.")

if __name__ == "__main__":
    main()
```

WEEK -11

25. Write a Python program using modular programming principles and demonstrate:

Input validation • Testing (minimum 3 test cases) • Debugging practice with comments



```
math_users
    src
        armstrong.py
        factorial.py
        input_validation.py
        main.py
        swap_numbers.py
```

In [ ]:

```
def is_armstrong(num):
    digits = str(num)
    power = len(digits)
    total = sum(int(d)**power for d in digits)
    return total == num

def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)

def swap_numbers(a, b):
    """
    Function to swap two numbers.
    Args:
        a (int): First number
        b (int): Second number
    Returns:
         Swapped values (a, b)
    """
    return b, a

# DEBUGGING PRACTICE:
# Print user choice to ensure correct menu selection
# print("Debug: User choice =", choice)

# DEBUGGING PRACTICE:
# Confirm function call based on user choice
# print("Debug: Calling selected module function")

# DEBUGGING PRACTICE:
# If program crashes, check import statements
# and function names

# DEBUGGING PRACTICE:
# Use try-except blocks to catch runtime errors
# and print meaningful error messgaes.



from src.armstrong import is_armstrong
from src.factorial import factorial
from src.swap_numbers import swap_numbers



def main():
    print ("welcome to math_users\n")

    while 'true':

        print("1.check armstrong number")
        print("2.factorial")
        print("3.swap")
        print("4.exit")

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


        if choice == "4":
            print("Exiting math_users...\n")
            break

        if choice == '1':
           num = int(input("enter a num : \n"))
           if is_armstrong(num):
              print(f"{num} is an armstrong num\n")
           else:
              print(f"{num} is not an armstrong num")
        elif choice == '2':
            num=int(input("enter a num: "))
            print("factorial:", factorial(num))
        elif  choice =='3':
             num1 = int(input("enter a num : "))
             print("a",a,"b",b)
        else:
            print("Invalid choice. Try again.")

if __name__ == "__main__":
    main()
```

WEEK -12

26. Write a Python project for a User Registration System with input validation,

testing, and debugging documentation.

Algorithm

1. Start program
2. Display registration form (ask for username, password, email, age)
3. Validate inputs:
4. Username: non-empty, alphanumeric
5. Password: minimum 8 chars, contains letters & numbers
6. Email: must contain @ and .
7. Age: must be integer ≥ 18
8. If validation fails → show error message & re-prompt
9. If validation passes → save user details to a file/database

10. Confirm successful registration
11. End program

In [8]:
```python
# User Registration System

def validate_username(username):
    if len(username) < 3:
        raise ValueError("Username must be at least 3 characters long")
    return True

def validate_email(email):
    if "@" not in email or "." not in email:
        raise ValueError("Invalid email format")
    return True

def validate_password(password):
    if len(password) < 6:
        raise ValueError("Password must be at least 6 characters long")
    return True

def validate_age(age):
    if age < 18:
        raise ValueError("User must be at least 18 years old")
    return True

def register_user():
    try:
        username = input("Enter username: ")
        validate_username(username)

        email = input("Enter email: ")
        validate_email(email)

        password = input("Enter password: ")
        validate_password(password)

        age = int(input("Enter age: "))
        validate_age(age)

        print("\n✅ Registration Successful!")
        print("User Details:")
        print("Username:", username)
        print("Email:", email)
        print("Age:", age)

    except ValueError as ve:
        print("\n❌ Error:", ve)
    except Exception as e:
        print("\n❌ Unexpected Error:", e)

# Main program
register_user()
```

```
✅ Registration Successful!
User Details:
Username: Hari priya
Email: Haripriya @gmai.com
Age: 18
```