# Loops : problems

- Count number of digits in an integer number.
- Find out sum of all digits of a given integer number.
- Find out the reverse of a given integer number.

- Display the first N terms of Fibonacci series.
- Check if the given integer number is Prime or not.
- Calculate the factorial of a given integer number.

# Lesson 3: Summary

Here is what we learned

- Control Statements
  - Decision Making
  - Loops

## Lesson 4 : topics

- Modular Programming
- Using Functions

# Lesson 4 : Modular Programming using Functions

# Modular Programming

- Modular programming is a methodology that encourages breaking down a program into smaller, independent modules or functions.
- It enhances code organization, readability, and maintainability.

# Functions

- Function is a block of code that encapsulates a specific task or related group of tasks.
- Functions are defined by a name, may have parameters and may return a value.
- Functions are executed only when it is called.

- Advantages:
  - **Modularity**
  - **Code Reusability**
  - **Readability**
  - **Debugging**
  - **Abstraction**
  - **Scoping**
  - **Testing**
  - **Collaboration**

# Sum of factorials of 3 numbers

Problem : Find sum of factorials of 3 given numbers

Algorithm:

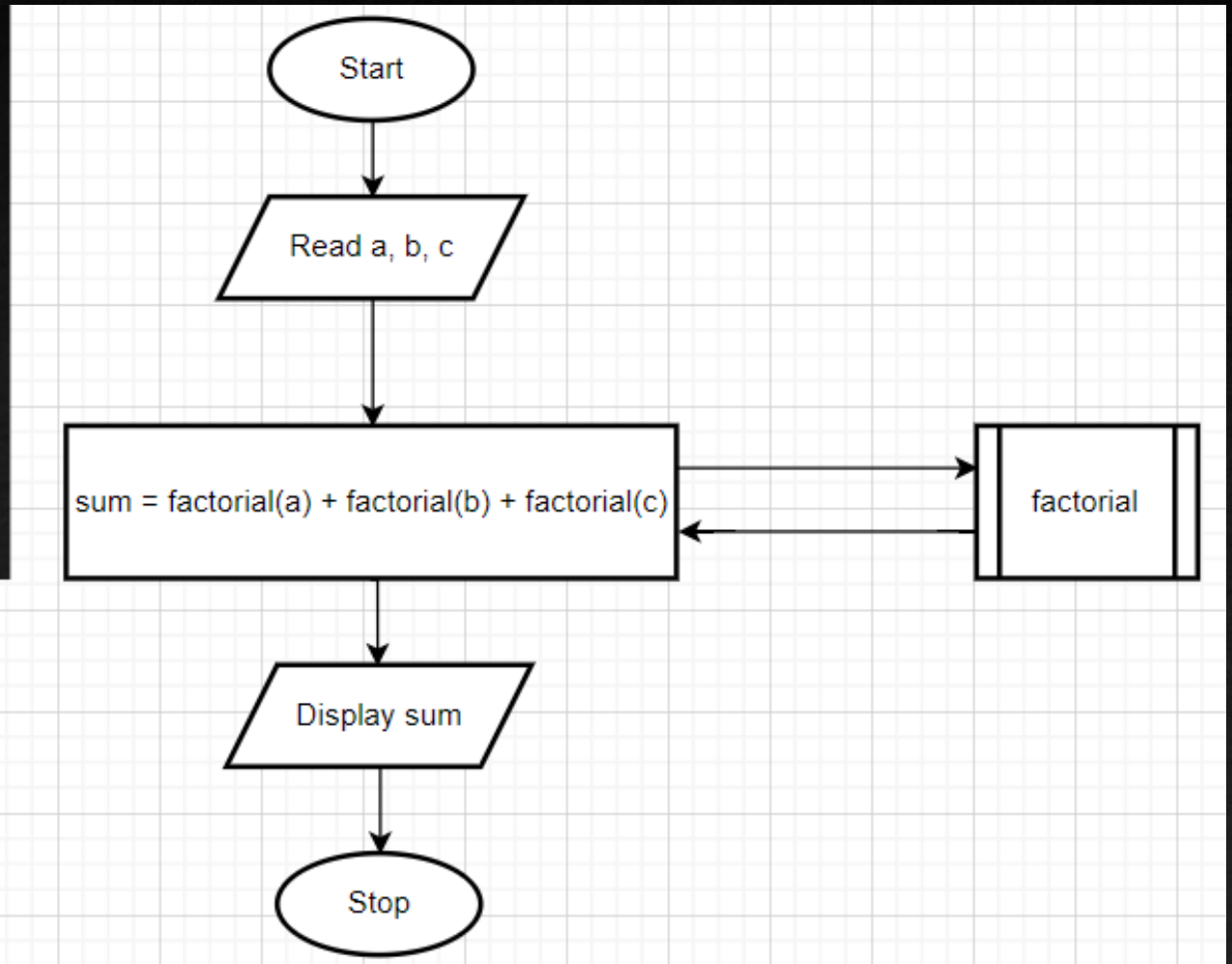
1. Start
2. Declare a, b, c, sum
3. Read a, b, c
4. sum = factorial(a) + factorial(b) + factorial(c)
5. Display sum
6. Stop

# Sum of factorials of 3 numbers

Problem : Find sum of factorials of 3 given numbers

Algorithm:

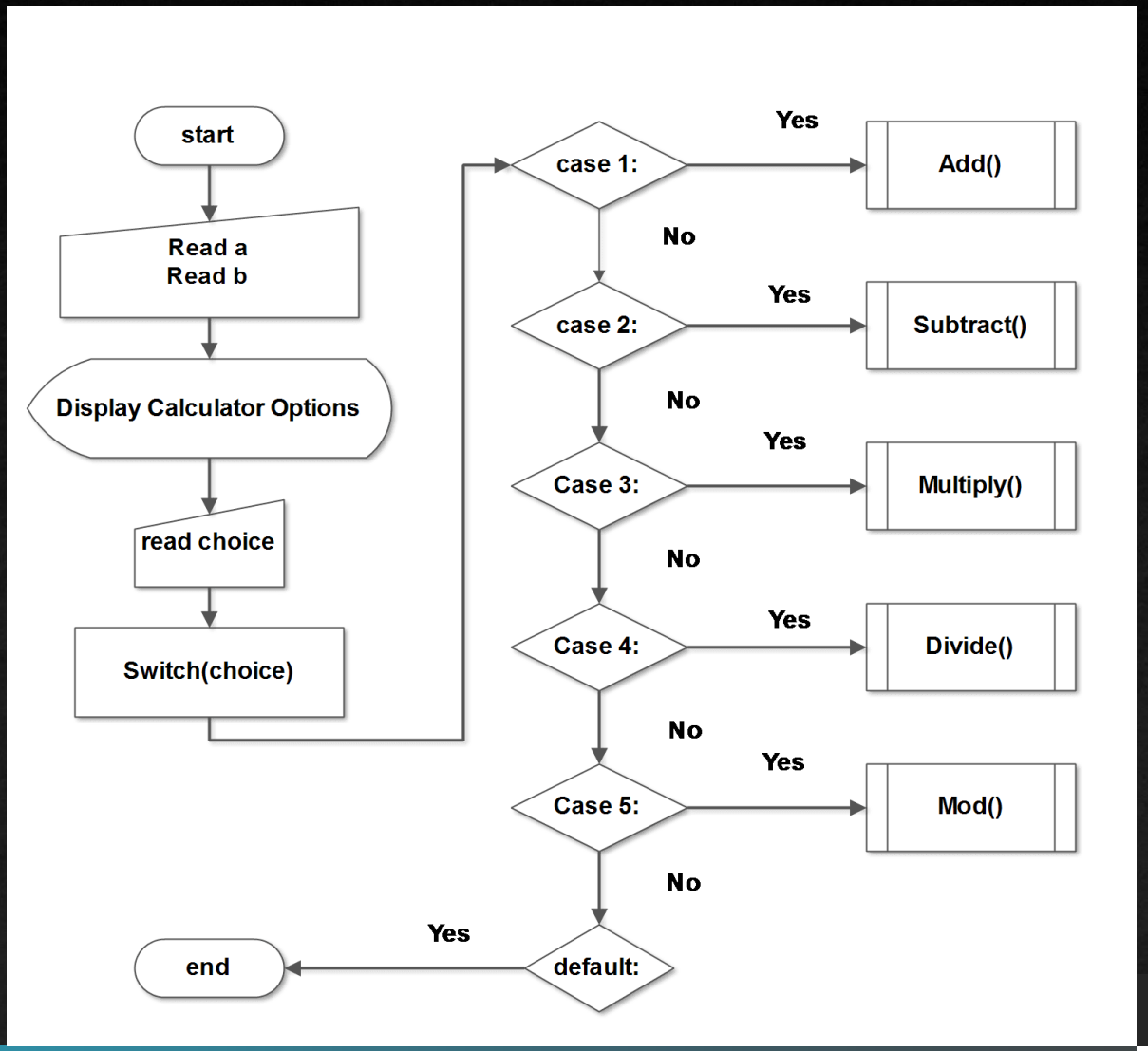
1. Start
2. Declare a, b, c, sum
3. Read a, b, c
4. sum = factorial(a) + factorial(b) + factorial(c)
5. Display sum
6. Stop



# Simple calculator using Functions

**Algorithm:**

1. Start
2. Read a, b
3. Display Calculator options
4. Read choice
5. Switch on choice
   - case 1 : add(a,b)
   - case 2 : subtract(a,b)
   - case 3 : multiply(a,b)
   - case 4 : divide(a,b)
   - case 5 : mod(a,b)
6. Stop



# Lesson 4: Summary

Here is what we learned

- Modular Programming using Functions