# Lab 11 Tasks

## Task 01

Create a custom exception class InvalidValueException that inherits from std::exception. Override the what() method to return a meaningful error message. Use this exception in a function validateAge(int age) that throws it if age is negative or greater than 120.

**Requirements:**

- Inherit from std::exception.
- Use const char* what() const noexcept override to provide error messages.
- Handle the exception in main().

**Sample Input/Output:**

Enter age: -5

Error: InvalidValueException - Age cannot be negative or exceed 120.

## Task 02

Create a template class Stack<T> with a fixed capacity. Implement methods push() and pop(). Throw:

- StackOverflowException (class) if pushing to a full stack.
- StackUnderflowException (class) if popping from an empty stack.

**Requirements:**

- Both exceptions inherit from std::exception and override what().
- Include error details (e.g., "Stack is full!").
- Demonstrate exceptions in main().

**Sample Input/Output:**

Pushing to a full stack: StackOverflowException - Stack is full!

Popping from an empty stack: StackUnderflowException - Stack is empty!

## Task 03

Create an exception hierarchy:

- Base class: FileException (with what()).
- Derived classes: FileNotFoundException, PermissionDeniedException.
  Write a function readFile(const string& filename) that throws these exceptions based on file issues.

**Requirements:**

- Use polymorphism to catch base class FileException.
- Handle specific exceptions in separate catch blocks.

**Sample Input/Output:**

Reading 'secret.txt': PermissionDeniedException - Access denied!

## Task 04

Write a template function sqrt(T num) that computes the square root. Throw:

- NegativeNumberException (class) if num is negative (for integers/doubles).
- InvalidTypeException (class) if T is not numeric (e.g., string).

**Requirements:**

- Use typeid or template specialization for type checks.
- Handle both exceptions in main().

**Sample Input/Output:**

sqrt(-4): NegativeNumberException - Input must be non-negative!

sqrt("hello"): InvalidTypeException - Non-numeric type detected!

# Task 05

Create a template class BankAccount<T> where T is the currency type (e.g., double, int). Throw an InsufficientFundsException (class) if a withdrawal exceeds the balance. Include the deficit amount in the exception message.

**Requirements:**

- withdraw(T amount) throws the exception if amount > balance.
- Catch the exception and display the deficit.

**Sample Input/Output:**

Balance: $500.00

Withdraw $600: InsufficientFundsException - Deficit: $100.00