Lab 10

Topics: Inheritance, Function Overriding, Exception Handing, File Input/Output and Template

Inheritance

1. **Single Inheritance**:

- In single inheritance, a derived class inherits from only one base class.

```
// Single Inheritance Example
class Base {
public:
    int data;
};

class Derived : public Base {
public:
    void display() {
        cout << "Data from base class: " << data << endl;
    }
};
```

2. **Multiple Inheritance**:

- Multiple inheritance allows a derived class to inherit from more than one base class.

```
// Multiple Inheritance Example
class Base1 {
public:
    int data1;
};

class Base2 {
public:
    int data2;
};

class Derived : public Base1, public Base2 {
public:
    void display() {
        cout << "Data from base 1: " << data1 << endl;
        cout << "Data from base 2: " << data2 << endl;
    }
};
```

3. **Multilevel Inheritance**:

- In multilevel inheritance, a derived class inherits from another derived class.

```
// Multilevel Inheritance Example
class Base {
public:
    int data;
};

class Derived1 : public Base {
public:
    // Some members
};

class Derived2 : public Derived1 {
public:
    void display() {
        cout << "Data from base class: " << data << endl;
    }
};
```

4. **Hierarchical Inheritance**:

- Hierarchical inheritance involves multiple derived classes inheriting from a single base class.

```
// Hierarchical Inheritance Example
class Base {
public:
    int data;
};

class Derived1 : public Base {
public:
    // Some members
};

class Derived2 : public Base {
public:
    // Some members
};
```

Exception handling

Exception handling is a mechanism in programming languages like C++ that deals with errors or exceptional situations that occur during the execution of a program. Instead of letting these errors crash the program or lead to undefined behavior, exception handling provides a structured way to handle them gracefully.

The key components of exception handling in C++ are:

1. **try**: This block encloses the code that might throw an exception.
2. **throw**: This statement is used to throw an exception explicitly when an error condition is encountered.
3. **catch**: This block catches and handles exceptions thrown within the corresponding try block.
4. **exception**: This is an object that represents an exceptional condition. It is typically derived from the standard exception class.

Templates

Templates in C++ allow you to write generic code that can work with any data type. They provide a way to create functions, classes, or structures that can operate with any data type without having to write separate implementations for each type. Templates are a powerful feature of C++ that enable code reusability and flexibility.

- **Function Template**
- **Function Template**

Home Practice

1. **Show the implementation of template class library for swap function**