sjsytu

**Learning Report – Intermediate C++ and Testing**

Course Code: <CODE>

./



Table of Contents

[Activity – 1 - Intermediate C++ and Testing 3](#_Toc59870413)

[**Goal** 3](#_Toc59870414)

[**Type of Activity – Individual** 3](#_Toc59870415)

[Set 1 - Classes and objects 3](#_Toc59870416)

[Set 2 - Operator overloading 3](#_Toc59870417)

[Set 3 – Inheritance 3](#_Toc59870418)

[Set 4 – Templates 4](#_Toc59870419)

[**GitHub Link** 4](#_Toc59870420)

[**Difficulties** 4](#_Toc59870421)

[Activity – 2 – Group Activity 4](#_Toc59870422)

[**Goal** 4](#_Toc59870423)

[**Type of Activity – Group activity** 5](#_Toc59870424)

[1. Inheritance 5](#_Toc59870425)

[2. Virtual functions 6](#_Toc59870426)

[3. Lists 6](#_Toc59870427)

[Activity – 3 – Individual Activity 7](#_Toc59870428)

[**Goal** 7](#_Toc59870429)

[**Type of Activity – Individual activity** 7](#_Toc59870430)

[References 8](#_Toc59870431)

# Activity – 1 - Intermediate C++ and Testing

## **Goal**

To gain more knowledge in Intermediate C++, to have better understanding in testing the code and improving code quality according to industry standards.

## **Type of Activity – Individual**

### Set 1 - Classes and objects

Class is a user defined data type, which holds its own data members and member functions, which can be accessed and used by creating instance of that class.

The variables inside class definition are called as data members and the functions are called member functions.

Objects are instances of class, which holds the data variables declared in class and the member functions work on these class objects.

Each object has different data variables. Objects are initialized using special class functions called **Constructors**. We will study about constructors later.

And whenever the object is out of its scope, another special class member function called **Destructor** is called, to release the memory reserved by the object.

### Set 2 - Operator overloading

It is a type of polymorphism in which an operator is overloaded to give user defined meaning to it. Overloaded operator is used to perform operation on user-defined data type. For example '+' operator can be overloaded to perform addition on various data types, like for Integer, String (concatenation) etc.

Operator overloading can be done by implementing a function which can be:

1. Member Function
2. Non-Member Function
3. Friend Function

### Set 3 – Inheritance

 The class whose properties are inherited by other class is called the **Parent** or **Base** or **Super** class. And, the class which inherits properties of other class is called **Child** or **Derived** or **Sub** class.

Purpose of Inheritance

1. Code Reusability
2. Method Overriding (Hence, Runtime Polymorphism.)
3. Use of Virtual Keyword

### Set 4 – Templates

The Standard Template Library (STL) is a set of C++ template classes to provide common programming data structures and functions such as lists, stacks, arrays, etc. It is a library of container classes, algorithms, and iterators. It is a generalized library and so, its components are parameterized. A working knowledge of template. Class is a prerequisite for working with STL.

**STL has four components**

* Algorithms
* Containers
* Functions
* Iterators

## **GitHub Link**

git@github.com:99003140/Intermediate-C-.git

## **Difficulties**

To write google test case for a few programs like that in templates and inheritance, but with the help facilitators, online resources and practice we were able to successfully complete the task.

# Activity – 2 – Group Activity

## **Goal**

To learn the concepts of Inheritance, Virtual Functions and Lists from STL.

## **Type of Activity – Group activity**

### Inheritance

 The class whose properties are inherited by other class is called the **Parent** or **Base** or **Super** class. And, the class which inherits properties of other class is called **Child** or **Derived** or **Sub** class.

Purpose of Inheritance in C++

1. Code Reusability
2. Method Overriding (Hence, Runtime Polymorphism.)
3. Use of Virtual Keyword

**Types of Inheritance**

1. Single level
2. Multi- level
3. Multiple
4. Hybrid
5. Hierarchy
6. Multi-path

Access Modifiers and Inheritance: Visibility of Class Members

Depending on Access modifier used while inheritance, the availability of class members of Super class in the sub class changes. It can either be private, protected or public.

**1)** **Public Inheritance**

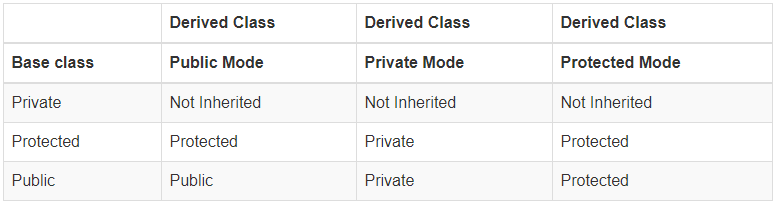
This is the most used inheritance mode. In this the protected member of super class becomes protected members of sub class and public becomes public.

**2) Private Inheritance**

In private mode, the protected and public members of super class become private members of derived class.

**3) Protected Inheritance**

In protected mode, the public and protected members of Super class becomes protected members of Sub class.



### Virtual functions

Virtual Function is a function in base class, which is override in the derived class, and which tells the compiler to perform **Late Binding** on this function.

Virtual Keyword is used to make a member function of the base class Virtual.

Important Points to Remember

1. Only the Base class Method's declaration needs the **Virtual** Keyword, not the definition.
2. If a function is declared as **virtual** in the base class, it will be virtual in all its derived classes.
3. The address of the virtual Function is placed in the **VTABLE** and the compiler uses **VPTR** (pointer) to point to the Virtual Function.

### 3. Lists

* Linked List is a very commonly used linear data structure which consists of group of nodes in a sequence.
* Each node holds its own data and the address of the next node hence forming a chain like structure.
* Linked Lists are used to create trees and graphs.

Advantages of Linked Lists

* They are a dynamic in nature which allocates the memory when required.
* Insertion and deletion operations can be easily implemented.
* Stacks and queues can be easily executed.
* Linked List reduces the access time.

Disadvantages of Linked Lists

* The memory is wasted as pointers require extra memory for storage.
* No element can be accessed randomly; it has to access each node sequentially.
* Reverse Traversing is difficult in linked list.

Few essential Methods in lists

emplace\_back 🡺 constructs an element in-place at the end.

pop\_back 🡺 removes the last element.

insert 🡺 inserts elements to the list.

push\_back 🡺 adds elements at the end.

# Activity – 3 – Individual Activity

## **Goal**

To have a hands on experience on STL concepts, string functions and test fixtures.

## **Type of Activity – Individual activity**

**Learning Outcomes**

* Used test fixtures to access functions without creating an object always and using the pointer to access the function in the class.
* Implemented various string operator overloading functions to implement operator overloading concepts.
* Learnt to implement various STL methods like – insert (), pushback (), emplace back ().
* Made use of the concept of iterator to traverse within a list or vector or any type of STL.
* **Programs executed** are – MySring operations, Library Book using STL and test fixtures on account programs.

# References

1. <https://en.cppreference.com/w/cpp/container>
2. <https://www.cplusplus.com/doc/tutorial/templates/>
3. <https://www.geeksforgeeks.org/initialize-a-vector-in-cpp-different-ways/?ref=leftbar-rightbar>
4. <https://www.learncpp.com/cpp-tutorial/introduction-to-operator-overloading/>