***Someshwar Shikshan Prasarak Mandal’s***

## Sharadchandra Pawar College Of Engineering and Technology, Someshwarnagar

**Object Oriented Programming**

**Laboratory Manual Computer Engineering**

Object Oriented Programming

**Laboratory Manual**

# Table of Contents

|  |  |  |  |
| --- | --- | --- | --- |
| Sr . No. | Topic | | Page. No. |
| 1. | **Vision, Mission** | |  |
| 2. | **How to Use This Manual** | |  |
| 3. | **PEOs and Pos** | |  |
| 4. | **Course Objective** | |  |
| 5. | **Laboratory Objective** | |  |
| 6. | **Experiment Learning Outcome (ELO)** | |  |
| 7. | **Lab Plan** | |  |
| **Group A** | | |  |
| **Sessions** | | |  |
| 1. | Implement a class Complex which represents the Complex Number | |  |
| data type. Implement the following operations: | |
| 1. | Constructor (including a default constructor which creates the |
| complex number 0+0i). | |
| 2. | Overloaded **operator+** to add two complex numbers. |
| 3. | Overloaded **operator\*** to multiply two complex numbers. |
| 4. | Overloaded << and >> to print and read Complex Numbers. |
| 2. | Develop an object oriented program in C++ to create a database of student information system containing the following information: Name, Roll number, Class, division, Date of Birth, Blood group, Contact address, telephone number, driving license no. etc Construct the database with suitable member functions for initializing and destroying the data viz constructor, default constructor, Copy constructor, destructor, static member functions, friend class, this pointer, inline code and dynamic memory allocation operators-new and delete. | |  |
|  | |
| 3. | Imagine a publishing company which does marketing for book and audio cassette versions. Create a class publication that stores the title (a string) and price (type float) of publications. From this class derive two classes: book which adds a page count (type int) and tape which adds a playing time in minutes (type float).  Write a program that instantiates the book and tape class, allows user to enter data and displays the data members. If an exception is caught, replace all the data member values with zero values | |  |
|  | |
|  | |

|  |  |  |
| --- | --- | --- |
| 4. | Write a C++ program that creates an output file, writes information to it, closes the file, open it again as an input file and read the information from the file. |  |
| 5. | Write a function template for selection sort that inputs, sorts and outputs an integer array and a float array. |  |
| 6. | Write C++ program using STL for sorting and searching user defined records such as personal records (Name, DOB, Telephone number etc) using vector container.  OR  Write C++ program using STL for sorting and searching user defined records such as Item records (Item code, name, cost, quantity etc) using vector container. |  |
| 7. | Write a program in C++ to use map associative container. The keys will be the names of states and the values will be the populations of the states. When the program runs, the user is prompted to type the name of a state. The program then looks in the map, using the state name as an index and returns the population of the state. |  |

To develop competent engineers to meet contemporary & emerging challenges for societal benefits.

**Mission**

1. To provide quality engineering education to the students through effective teaching learning process.
2. To strengthen skills and competencies for resolving interdisciplinary problems by providing conducive environment.
3. To provide opportunity for dealing with real life problems through industry institutes interaction.

# How to Use This Manual

This Manual assumes that the facilitators are aware of Collaborative Learning Methodologies.

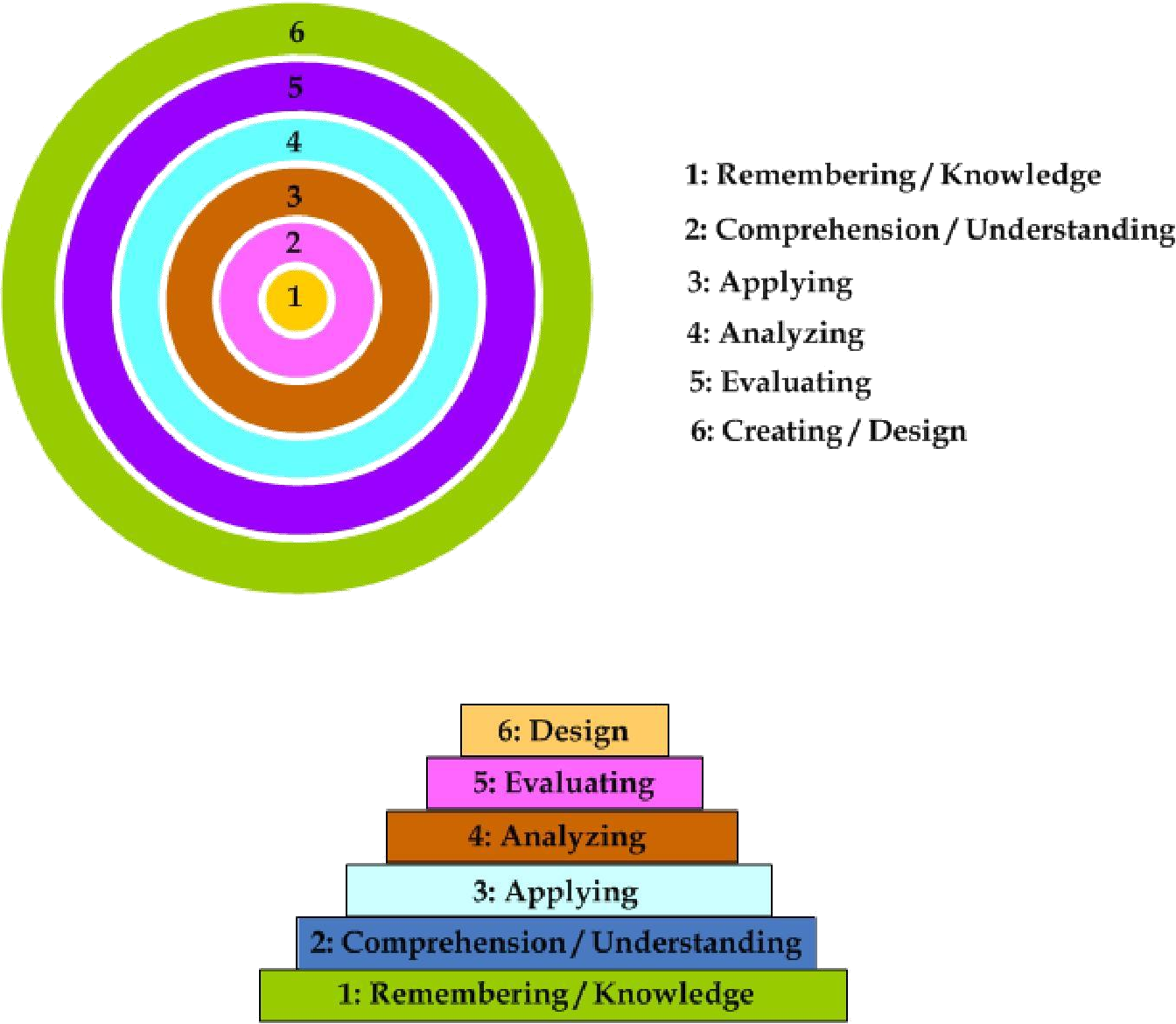
This Manual will only provide them tool they may need to facilitate the session on Computer Organization module in collaborative learning environment.

The Facilitator is expected to refer this Manual before the session.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **K** |  |  |  |  |
| Applying | Problem | | Design & | Investigation |
| Knowledge | Analysis | | Development | of problems |
| (PO:a) | (PO:b) | | (PO:c) | (PO:d) |
|  |  |  |  |  |
|  | **E** | | **E** | **T** |
| Modern | Engineer & | | Environment | Ethics |
| Tool | Society | | Sustainability | (PO:i) |
| Usage | (PO:f) | | (PO:h) |  |
| (PO:e) |  |  |  |  |
| **T** | **O** | | **M** | **I** |
| Individual | Communication | | Project | Life Long |
| & Team | (PO:k) | | Management | Learning |
| work |  |  | & Finance | (PO:l) |
| (PO:g) | (PO:j) |  |



**Disk Approach- Digital Blooms Taxonomy**



#### Experiment No. 1:

**Title :** Implement a class complex which represent the Complex Number data type . Implement the following operations:

* 1. Constructor (include a default constructor which creates the complex number 0+0i )
  2. Overloaded operator + to add two polynomials of degree 2.
  3. Overloaded operator \* to multiply two complex number.
  4. Overloaded << and >> to print and read Complex Number.

**Aim :** To implement a class complex which represent the Complex Number data type

#### Theory :

Concept of overloading

There are two ways of overloading that are operator overloading & function overloading . here in this program we use operator overloading .

Operator overloading -

1. operator overloading can be defined a new meaning for an existing "operator" . There are many operator which can be overloaded such as mathematical operator, logical operator , relational operator etc.
2. There are some operators which cannot be overloaded, that are
   1. . And \*
   2. size of
   3. : : (scope resolution )
   4. : : ? (conditional operator )
3. C++ provide the flexibility to the programmers in extending these build-in operators .

Need of operator overloading -

There are predefined operators such as + , - ,\* , / & so on which operates on fundamental data types such as integer , double , char and so on. In order to make user defined data type as natural as fundamental data type, the user defined data type can be associate with the set of predefined operators the concept called operator over loading is used .

Note that the fundamental meaning of these operators is not at all changed by operator overloading

.Rather; with this meaning these operators are associated with the user defined data type. Rules for operator overloading –

1. The basic meaning of operator cannot be changed.
2. Binary arithmetic operator (\* , - ,+, / ) must return a value.
3. Binary operator overloaded through member function must take one explicite argument . 4.Binary operator overloaded through friend function takes two arguments

#### Conclusion :

By doing above program I conclude that operators can be overloaded to do the operations on objects of class complex.

**Experiment No. 2 :**

**Title :**

Develop an object oriented program in C++ to create a database of student information system containing the following information : Name, Roll number, Class, Division, Date of Birth, Blood group, Contact address, Telephone number, Driving license no. etc Construct the database with suitable member functions for initializing and destroying the data viz Constructor, Default constructor, Copy constructor,Destructor, Static member functions, friend class, this pointer, inline code and dynamic memory allocation operators-new and delete .

### Aim :

To use object oriented concepts for implementation of student database .

### Theory :

Constructor :

It is special member function whose task to initialize of object of its class. Its name is same as class name & the constructor is invoked whenever an object of its associated class is created. It does not have any return type. It is called constructor because it construct the values of data member of the class.

Default constructor :

When no arguments will be passed through the constructor , it becomes a

default constructor. This is used to create objects & initialize them.

Copy constructor : A constructor can accept a reference to its own class as a parameter. Thus, the statement

class shape

{

public:

shape(shape &);

};

In such cases, the constructor is called the copy constructor. A copy constructor is used to declare & initialize an object from another object.

Destructor :

Destructor is a special member function of class i.e. executed whenever object of

this class goes out of the scope or delete expression in that apply to the pointer of object of this class. It uses tilde operator (~).

A destructor never takes any arguments nor does it return any value.

Static Member Functions :

properties:

A member function that is declared static has the following

* 1. A static function can have access to only static members ( functions or variables ) declared in the same class.
  2. A static member function can be called using the class name ( instead of its objects) as

follows:

class name : : function name ;

Friend Class:

A class can have friendship with another class, if there are two classes first & second. If first class grants friendship with other class second then the private data member of the class first are permitted to be access by the public member of the class second but on other hand public member of class first can not access the private member of the class second.

this pointer :

this is pointer that points to the object for which this function was called. This unique pointer is automatically passed to a member function when it is called. The pointer this acts as an implicit argument to all the member function.

class ABC

{

int a ;

};

C++ uses a unique keyword called this to represent an object that invokes a member function this is a pointer that points to the objects for which this function was called . For example, the function call , A max () will set the pointer this to the address of the object A . The starting address is the same as the address of the first variable in the class structure .

This unique pointer is automatically passed to a member function when it is called . The pointer this acts as an implicit argument to all the member functions . Consider the following simple example :

class ABC

{

.....int a ;

.................

..................

};

The private variable 'a' can be used directly inside a member function , like a = 123 ;

We can also use the following statement to do the same job : this->a = 123 ;

Since C++ permits the use of shorthand form a =123 , we have been using the pointer this explicitly so far . However , we have been implicitly using this pointer this when overloading the operators using member function .

Inline code :

C++ has a different solution to this problem . To eliminate the cost of calls to small functions , c++ proposes a new feature called inline function . An inline function is a function that is expanded in line when it is invoked. That is, the compiler replaces the function call with the corresponding function code

.

e.g.

inline function name

{

function body;

}

Dynamic Memory Allocation Operators :

new & delete :

When new is used to allocate memory for a C++ class object, the object's constructor is called after the memory is allocated. Use the delete operator to deallocate the memory allocated with the new operator.

.

### Conclusion :

By executing this program ,I understand the concepts of constructor ,destructor , new , delete, inline function , static function, static variables, copy constructors etc.

#### Experiment No. 3 :

**Title :**

Imagine a publishing company which does marketing for book and audiocassette versions. Create a class publication that stores the title (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing time in minutes (type float).

Write a program that instantiates the book and tape classes, allows user to enter data and displays the data members. If an exception is caught, replace all the data member values with zero values.

#### Aim:

To implement a class which stores the title and price of book and its two derived classes to store page count and play time also to handle exception when zero value for data member occurs.

#### Theory :

Inheritance:

Base Class and Derived Class:

The class from which the data members and member functions are used by another class is called the base class.

The class which uses the properties of base class and at the same time can add its own properties is called derived class.

There are three types of access specifier or qualifier using which the members of the class are accessed by the other class –

* 1. Private 2.Public 3.Protected

If base class has private members then those members are not accessible to derived class.Protected members are public to derived classes but private to rest of the program.Public members are accessible to all.

The derived class can inherit base class publicly or privately. The notation used for inheritance is: For ex:

class derived\_class\_name : access\_specifier base\_class\_name

The first line indicates that there are two classes d1 and b1.It means “ the derived class d1 inherits the base class b1 publicly” .

The second line indicates that there are two classes d2 and b2.It means “ the derived class d2 inherits the base class b2 privately” .

The base class and derived class can generate their own objects.These objects differ from each other. Relation between Base class and Derived class:

Inheritance is an important feature in object oriented programming that allows the re usability of the code.

The fundamental idea behind the inheritance is that – make use of data members and member functions of base class in derived class along with some additional fundamentalism present in derived class.

Constructor and Destructor in Derived Class:

When we create an object for derived class then first of all the Base class constructor is called and after that the derived class constructor is called.

The reason behind this is that is that the Derived class inherits from the Base class,both the Base class and Derived class constructors will be called when a Derived class object is created.

This is also called as chain of constructor calls.

Exception Handling:

Exception are of two kinds namely, synchronous exception and asynchronous exception.

The errors that are caused by events beyond the control of the program (such as keyboard interrupts )are called asynchronous exception. The proposed exception handling mechanism in c++ is designed to handle only synchronous exception.

Exception Handling mechanism:

C++ exception handling mechanism is basically built upon three keywords namely ,try , throw and catch.

The keyword try is used to preface a block of statements surrounded by braces which may generate exceptions. These block of statements are known as try block. When expression is detected it is known as throw statement.

A handler may decide to rethrow the exception caught without processing it. In such situations , we may simply invoke throw without any argument as shown below:

Throw;

This causes the current exception to be thrown to the next enclosing try/catch sequence and is caught by a catch statement listed after that enclosing try block.

#### Conclusion:

From this program we can conclude that by using inheritance and exception handling we can reduce the efforts in writing the code. Inheritance helps us to use a particular member again without any need of writing it again while exception handling helps to detect the exceptions that occur during runtime.

#### Experiment No. 4 :

**Title :**

Write a C++ program that creates an output file, writes information to it, closes the file and open it again as an input file and read the information from the file.

#### Aim :

To create an output file writes information to it closes the file and open it again as an input file and read the information from the file.

#### Theory:

Stream class hierarchy :

The stream can represent file, console , block of memory or hardware device .the iostream library provides the common set of functions for handling these streams .

Open file operation :

The file operations are associated with the object of the classes ifstream , ofstream or fstream. Hence we need to create an object of corresponding classes .

the file can be opened by the function called open() . The syntax of file open is open(file\_name,mode);

The file name is a null terminated string that represents the name of the file that is to be open . we can use open() function using the given syntax as :

ofstream obj ; obj.open("sample.dat",ios::out|ios::binary);

That means the file sample.dat is opened for output operation in binary mode. Thus we can combine the flags using OR( | ) operator.

This is\_open() is a boolean function that can be used to check whether the file is open or not. for example

if(obj.is\_open())

{

cout<<” File is successfully opened for operations” ;

}

Close file operation :

To close the file the member function close() is used. The close function takes no parameter and returns no value.

object.close();

You can detect when the end of an input file has been reached by using the eof() member function of ios. It returns true when the end of the file has been encountered and false otherwise.

Finding the End of the File

For finding end of the file we use eof() function. This function is a member function of ios class. If end of file I encountered then it return a non zero value.

For example if(file.eof()!=0)

{

cout<<” You are at the end of the file” ;

}

Unformatted I/O Functions :

Get and put functions:

The get and put function s are u s.ed to read and di s.play the content s. The syntax of get and put function s i s

Get(char ch);

Put(char ch);

Getline and write functions :

The getline function is used to read the file line by line and the write statement is used to write the contents either to the file or to the console.

File pointers:

File pointers are used for locating the position in the file. With each file object there are two pointers associated with it. The get pointer and put pointer are the pointers basically return the current get position and current put positions.

While performing file operations, we must be able to reach at any desired position inside the file. For this purpose there are two commonly used functions-

Seek

The seek operation is using two functions seekg and seekp.

Seekg means get pointer of specific location for reading the records. Seekp means get pointer of specific location for writing the records. The syntax of seek is

Seekg(offset, reference-position); Seekp(offset, reference-position);

Where, offset is any constant specifying the location, reference position is for specifying beginning, end of current position. It can be specified as:

ios::beg for beginning location

ios::end for end of file ios::cur for current location Tell :-

This function tells us the current position. For example

File.tellg() :- gives current position of get pointer (for reading the record)

File.tellp() :- gives current position of put pointer (for writing the record)

#### Conclusion :

By executing above program I understand that how the file is created so as to store the contains of any object permanently. It concludes that we can write the contains in the file and read that contains whenever require.

#### Experiment No. 5 :

**Title:**

Write a function template selection sort . write a program that includes sorts and outputs an integer array and float array.

#### Aim :

To implement generic programming using template keyword .

#### Theory :

Selection Sorting :

In computer science, selection sort is a sorting algorithm, specifically an in-place comparison sort. It has O(n2) time complexity, making it inefficient on large lists, and generally performs worse than the similar insertion sort. Selection sort is noted for its simplicity, and it has performance advantages over more complicated algorithms in certain situations, particularly where auxiliary memory is limited.

The algorithm divides the input list into two parts: the sublist of items already sorted, which is built up from left to right at the front (left) of the list, and the sublist of items remaining to be sorted that occupy the rest of the list. Initially, the sorted sublist is empty and the unsorted sublist is the entire input list. The algorithm proceeds by finding the smallest (or largest, depending on sorting order) element in the unsorted sublist, exchanging (swapping) it with the leftmost unsorted element (putting it in sorted order), and moving the sublist boundaries one element to the right

Template :

Templates are the foundation of generic programming, which involves writing code in a way that is independent of any particular type.

A template is a blueprint or formula for creating a generic class or a function. The library containers like iterators and algorithms are examples of generic programming and have been developed using template concept.

There is a single definition of each container, such as vector, but we can define many different kinds of vectors for example, vector <int> or vector <string>.

You can use templates to define functions as well as classes, let us see how do they work: Function Template:

The general form of a template function definition is shown here:

template <class type> ret-type func-name(parameter list)

{

// body of function

}

Here, type is a placeholder name for a data type used by the function. This name can be used within the function definition.

Class Template :

Just as we can define function templates, we can also define class templates. The general form of a generic class declaration is shown here:

template<class type>classclass-name

{

.. . . . . . . .

}

Here, type is the placeholder type name, which will be specified when a class is instantiated. You can define more than one generic data type by using a comma-separated list.

#### Conclusion :

By executing this program I understand the concept of generic programming.

#### Experiment No. 6 :

**Title:**

Write C++ program using STL for sorting and searching user defined records such as personal records (Name, DOB, Telephone number etc) using vector container.

**OR**

Write C++ program using STL for sorting and searching user defined records such as Item records (Item code, name, cost, quantity etc) using vector container.

#### Aim :

#### To learn the concept STL, searching, sorting and vector container.

#### Theory :

**STL:**

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 classes](https://www.geeksforgeeks.org/templates-cpp/) is a prerequisite for working with STL.

**STL has four components**

* Algorithms
* Containers
* Functions
* Iterators

**Algorithms**

* The algorithm defines a collection of functions especially designed to be used on ranges of elements.They act on containers and provide means for various operations for the contents of the containers.
* Algorithm
  + [Sorting](http://quiz.geeksforgeeks.org/sort-algorithms-the-c-standard-template-library-stl/)
  + [Searching](http://quiz.geeksforgeeks.org/binary-search-algorithms-the-c-standard-template-library-stl/)
  + [Important STL Algorithms](https://www.geeksforgeeks.org/c-magicians-stl-algorithms/)
  + [Useful Array algorithms](https://www.geeksforgeeks.org/useful-array-algorithms-in-c-stl/)
  + [Partition Operations](https://www.geeksforgeeks.org/stdpartition-in-c-stl/)
  + Numeric

**STL:**

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 classes](https://www.geeksforgeeks.org/templates-cpp/) is a prerequisite for working with STL.

**STL has four components**

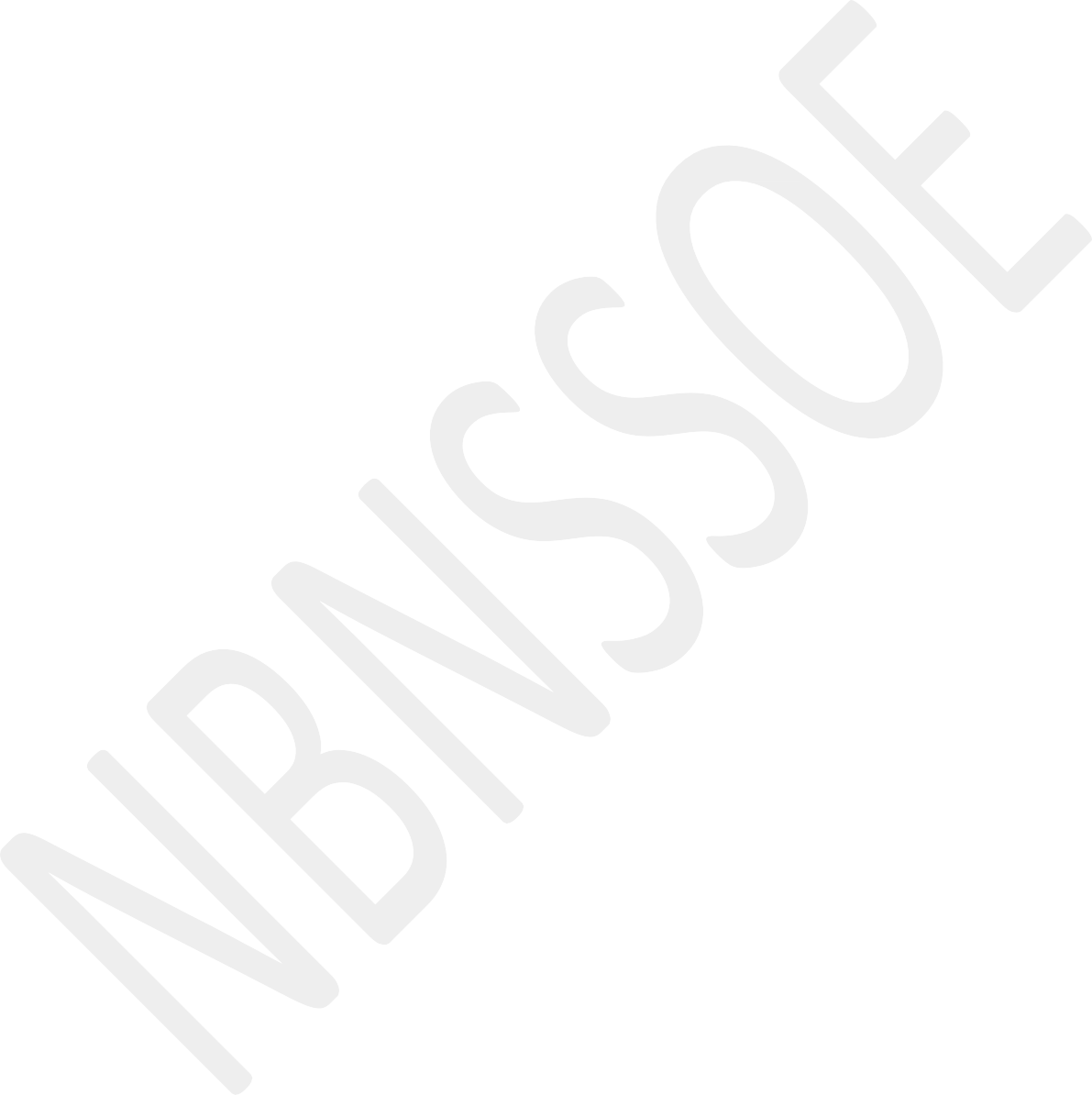
* Algorithms
* Containers
* Functions
* Iterators

**Algorithms**

* The algorithm defines a collection of functions especially designed to be used on ranges of elements.They act on containers and provide means for various operations for the contents of the containers.
* Algorithm
  + [Sorting](http://quiz.geeksforgeeks.org/sort-algorithms-the-c-standard-template-library-stl/)
  + [Searching](http://quiz.geeksforgeeks.org/binary-search-algorithms-the-c-standard-template-library-stl/)
  + [Important STL Algorithms](https://www.geeksforgeeks.org/c-magicians-stl-algorithms/)
  + [Useful Array algorithms](https://www.geeksforgeeks.org/useful-array-algorithms-in-c-stl/)
  + [Partition Operations](https://www.geeksforgeeks.org/stdpartition-in-c-stl/)
  + Numeric

**Containers**

* Containers or container classes store objects and data. There are in total seven standard “first-class” container classes and three container adaptor classes and only seven header files that provide access to these containers or container adaptors.
  + Sequence Containers: implement data structures which can be accessed in a sequential manner.
    - [vector](http://quiz.geeksforgeeks.org/vector-sequence-containers-the-c-standard-template-library-stl-set-1/)
    - [list](http://quiz.geeksforgeeks.org/list-sequence-containers-the-c-standard-template-library-stl/)
    - [deque](http://quiz.geeksforgeeks.org/deque-sequence-containers-the-c-standard-template-library-stl/)
    - [arrays](https://www.geeksforgeeks.org/array-class-c/)
    - [forward\_list](https://www.geeksforgeeks.org/forward-list-c-set-1-introduction-important-functions/)( Introduced in C++11)



* + Container Adaptors : provide a different interface for sequential containers.
    - [queue](http://quiz.geeksforgeeks.org/queue-container-adaptors-the-c-standard-template-library-stl/)
    - [priority\_queue](http://quiz.geeksforgeeks.org/priority-queue-container-adaptors-the-c-standard-template-library-stl/)
    - [stack](http://quiz.geeksforgeeks.org/stack-container-adaptors-the-c-standard-template-library-stl/)
  + Associative Containers : implement sorted data structures that can be quickly searched (O(log n) complexity).
    - [set](http://quiz.geeksforgeeks.org/set-associative-containers-the-c-standard-template-library-stl/)
    - [multiset](http://quiz.geeksforgeeks.org/multiset-associative-containers-the-c-standard-template-library-stl/)
    - [map](http://quiz.geeksforgeeks.org/map-associative-containers-the-c-standard-template-library-stl/)
    - [multimap](http://quiz.geeksforgeeks.org/multimap-associative-containers-the-c-standard-template-library-stl/)
  + Unordered Associative Containers : implement unordered data structures that can be quickly searched
    - [unordered\_set](https://www.geeksforgeeks.org/unordered_set-in-cpp-stl/)
    - [unordered\_multiset](https://www.geeksforgeeks.org/unordered_multiset-and-its-uses/)
    - [unordered\_map](https://www.geeksforgeeks.org/unordered_map-in-cpp-stl/)
    - [unordered\_multimap](https://www.geeksforgeeks.org/unordered_multimap-and-its-application/)

### Functions

* The STL includes classes that overload the function call operator. Instances of such classes are called function objects or functors. Functors allow the working of the associated function to be customized with the help of parameters to be passed.

### Iterators

* As the name suggests, iterators are used for working upon a sequence of values. They are the major feature that allow generality in STL.

### Utility Library

* Defined in header <utility>.
  + [pair](http://quiz.geeksforgeeks.org/pair-simple-containers-the-c-standard-template-library-stl/)

### Sorting:

It is one of the most basic functions applied to data. It means arranging the data in a particular fashion, which can be increasing or decreasing. There is a builtin function in C++ STL by the name of sort(). This function internally uses IntroSort. In more details it is implemented using hybrid of QuickSort,

HeapSort and InsertionSort.By default, it uses QuickSort but if QuickSort is doing unfair partitioning and taking more than N\*logN time, it switches to HeapSort and when the array size becomes really small, it switches to InsertionSort. The prototype for sort is :

### Searching:

It is a widely used searching algorithm that requires the array to be sorted before search is applied. The main idea behind this algorithm is to keep dividing the array in half (divide and conquer) until the element is found, or all the elements are exhausted.

It works by comparing the middle item of the array with our target, if it matches, it returns true

otherwise if the middle term is greater than the target, the search is performed in the left sub-array. If the middle term is less than target, the search is performed in the right sub-array.

The prototype for binary search is :

binary\_search(startaddress, endaddress, valuetofind)

startaddress: the address of the first element of the array.

endaddress: the address of the last element of the array.

valuetofind: the target value which we have to search for.

**//Searching:**

#include<iostream>

#include<string>

Using namespace std;

voids how(int a[], int arraysize)

{

for(inti = 0; i <arraysize; ++i) cout<< a[i] << " ";

}

intmain()

{

inta[] = { 1, 5, 8, 9, 6, 7, 3, 4, 2, 0 };

intasize = sizeof(a) / sizeof(a[0]); cout<< "\n The array is : "; show(a, asize);

cout<< "\n\nLet's say we want to search for 2 in the array"; cout<< "\n So, we first sort the array";

sort(a, a + asize);

cout<< "\n\n The array after sorting is : "; show(a, asize);

cout<< "\n\nNow, we do the binary search"; if(binary\_search(a, a + 10, 2))

cout<< "\nElement found in the array"; else

cout<< "\nElement not found in the array";

cout<< "\n\nNow, say we want to search for 10"; if(binary\_search(a, a + 10, 10))

cout<< "\nElement found in the array"; else

cout<< "\nElement not found in the array";

return0;

}

**Output:**

The array is : 1 5 8 9 0 6 7 3 4 2 0

Let's say we want to search for 2 in the array

So, we first sort the array

The array after sorting is : 0 1 2 3 4 5 6 7 8 9 Now, we do the binary search

Element found in the array

Now, say we want to search for 10 Element not found in the array

**Facilities:**

Linux Operating Systems, G++

**Algorithm:**

1. Start.
2. Give a header file to use ‘vector’.
3. Create a vector naming ‘personal\_records’.
4. Initialize variables to store name, birth date and telephone number.
5. Using iterator store as many records you want to store using predefined functions as push\_back().
6. Create another vector ‘item\_record’
7. Initialize variables to store item code, item name, quantity and cost.
8. Using iterator and predefined functions store the data.
9. Using predefined function sort(), sort the data stored according to user requirements.
10. Using predefined function search, search the element from the vector the user wants to check.
11. Display and call the functions using a menu**.**
12. End.

|  |
| --- |
| **Input:** |
| Personnel information such as name, DOB, telephone number. |
| **Output:** |

Display personnel information from database. The result in following format:

\*\*\*\*\* Menu \*\*\*\*\* 1.Insert

2.Display 3.Search 4.Sort 5.Delete 6.Exit

Enter your choice:1 Enter Item Name: bat Enter Item Quantity:2 Enter Item Cost:50

Enter Item Code:1

**Conclusion :**

Hence, we have successfully studied the concept of STL(Standard Template Library) and how it makes many data structures easy. It briefs about the predefined functions of STL and their uses such a search() and sort().

#### Experiment No. 7 :

**Title:**

Write a program in C++ to use map associative container. The keys will be the names of states and the values will be the populations of the states. When the program runs, the user is prompted to type the name of a state. The program then looks in the map, using the state name as an index and returns the population of the state.

#### Aim :

To use map associative container.

#### Theory :

Map associative container: Map associative container are associative containers that store elements in a mapped fashion. Each element has a key value and a mapped value. No two mapped values can have same key values.

**map::operator[]**

This operator is used to reference the element present at position given inside the operator. It is similar to the at() function, the only difference is that the at() function throws an out-of-range exception when the position is not in the bounds of the size of map, while this operator causes undefined behaviour.

**Syntax :**

**mapname[key]**

**Parameters :**

Key value mapped to the element to be fetched.

**Returns :**

Direct reference to the element at the given key value.

**Examples:**

Input : map mymap;

mymap['a'] = 1; mymap['a'];

Output : 1

Input : map mymap;

mymap["abcd"] = 7;

mymap["abcd"];

Output : 7

**//Program** :

#include <map>

#include<iostream>

#include<string>

Using namespace std;

int main()

{

// map declaration map<int,string>mymap;

// mapping integers to strings mymap[1] = "Hi";

mymap[2] = "This";

mymap[3] = "is";

mymap[4] = "NBN";

// using operator[] to print string

// mapped to integer 4 cout<<mymap[4]; return0;

}

**Output:**

#### NBN

**Facilities:**

Linux Operating Systems, G++

**Algorithm:**

1. Start.
2. Give a header file to map associative container.
3. Insert states name so that we get values as population of that state.
4. Use populationMap.insert().
5. Display the population of states**.**
6. End

|  |
| --- |
| **Input:** |
| Information such as state name to map associative container. |
| **Output:** |
| Size of population Map: 5 Brasil: 193 million  China: 1339 million  India: 1187 million  Indonesia: 234 million  Pakistan: 170 million  Indonesia's populations is 234 million |

#### Conclusion :

#### Hence, we have successfully studied the concept of map associative container