Experiment No-4

19UCS122

Title: Write a program to implement class in C++

Objectives:

1. To understand the concept of class and object
2. To understand the concept of constructor & destructor

key concepts: class, object, constructor, destructor

* Theory:

In object oriented design (OOD), the first step is to identify the components, called objects. An object combines data and the operations on that data in a single unit.

In C++, the mechanism that allows you to combine data and the operations on that data in a single unit is called a class. A class is a collection of a fixed number of components. The components of a class are called member of class.

* The general syntax for defining a class is

class

```
class classidentifier
{    classmemberlist;
};
```

in which classmemberlist consists of variable declarations and /or functions. That is, a member of a class can be either a variable (to store data) or a function

(1) If a member of a class is a variable, you declare it just like any other variable. Also, in the defination of the class, you cannot initialize a variable when you decbrei

(2) If a member of a class is a function, you typically use the function prototype to declare that member

The members of a class are classified into 3 catagories: Private, Public and protected.

19UCS122

Page No.:
Date:

In C++, Private, public and protected are reserved words and are called member access ~~specification~~ specifiers.

* Following are some facts about private and public members of a class.
  1. By default, all members of a class are private.
  2. If a member of a class is private, you cannot access it outside of class.
  3. ~~A~~ public member is accessible outside of class.
  4. ~~A~~ To make a member of a class public, you use the member access specifier public with a colon:

* Accessing class members
  Once an object of a class is declared, it can access the members of the class. The general syntax for an object to access a member of a class is:

  classObjectName.memberName

* The class members that a class object can access depend on where the object is declared.
  1. If the object is declared in the defination of a member function of the class, then the object can access both the public and private members
  2. If the object is declared elsewhere (for exampe in a user's program), then the object can access only the public members of the class.

In C++, the dot,(period), is an operator called the member access operator.

19UCS122

* Class scope

A class object can be either automatic(that is, created each time the control reaches its declaration and destroyed when the control exits the surrounding block) or static (that is, created once, when the control reaches its declaration, and destrored when the program terminates).

Also, you can declare an array of class objects. A class object has the same scope as other variables. A member of class has the same scope as a member of a struct. That is, a member of a class is local to the class. You access a class member outside of the class by using the class object name and the member access operator (.).

* Functions and Classes

Following rules describe relationship between functions and classes:

1. Class object can be passed as parameters to functions and returned as function values
2. As parameters to functions, class object can be passed either by value or by referance
3. If a class object is passed by value, the contents of member variables of actual parameter are copied into the corresponding member variables of the formal parameter.

* Constructions and Destructions :

Constructors and destructors are two special kinds of member functions. In general, a constructor is a member function with the same name as the class. A constructer is the first method that is called

Page No.:
Date:

$\boxed{19UCS122}$

implicitly when an object is created. They are used to initialize data and provide the guarantee that the data is always valid.

A destructor is the method that is called each time the object dies or exceeds its lifetime. It has the same name as the name of the class, prefixed with a ~ (character tilde). They are used to perform any cleanup activity for data members whose memory is allocated dynamically.

Ex: myClassA(); //constructor
~myclassA(); // destructor

constructors and destructors need to have public scope. Constructors can be overloaded, to support different ways of object initialization.