Assignment 8

Problem Statement

Create student database with appropriate data members that should use the following features of object oriented programming in C++. Class, Object, array of objects, new, delete, default constructor to initialize student class fields, parameterized constructor to set the values into the objects, access specifiers, this pointer.

Objectives:

To learn the concept of Class, Object, array of objects, new, delete, default constructor to initialize student class fields, parameterized constructor to set the values into the objects, access specifiers, this pointer.

Outcomes:

The performer will be able to design a program in c++ using concept of Class, Object, array of objects, new, delete, default constructor to initialize student class fields, parameterized constructor to set the values into the objects, access specifiers, this pointer.

Software Requirements:

1. 64-bit Open source Linux or its derivative
2. Open Source C++ Programming tool like G++/GCC.

Theory:

Constructors:

A special method of the class that will be automatically invoked when an instance of the class is created is called as constructor. Following are the most useful features of constructor.

1) Constructor is used for initializing the values to the data members of the Class.
2) Constructor is that whose name is same as name of class.
3) Constructor gets automatically called when an object of class is created.
4) Constructors never have a Return Type even void.

5) Constructor is of Default, Parameterized and Copy Constructors.

The various types of Constructor are as follows:-
Constructors can be classified into 3 types
1. Default Constructor
2. Parameterized Constructor
3. Copy Constructor

1. **Default Constructor**:- Default Constructor is also called as Empty Constructor which has no arguments and It is Automatically called when we creates the object of class but Remember name of Constructor is same as name of class and Constructor never declared with the help of Return Type. Means we can‟t declare a Constructor with the help of void Return Type. , if we never Pass or declare any Arguments then this called as the Copy Constructors.

2. **Parameterized Constructor**: - This is another type constructor which has some Arguments and same name as class name but it uses some Arguments So For this We have to create object of Class by passing some Arguments at the time of creating object with the name of class. When we pass some Arguments to the Constructor then this will automatically pass the Arguments to the Constructor and the values will retrieve by the Respective Data Members of the Class.

3. **Copy Constructor**: - This is also another type of Constructor. In this Constructor we pass the object of class into the Another Object of Same Class. As name Suggests you Copy, means Copy the values of one Object into the another Object of Class .This is used for Copying the values of class object into an another object of class So we call them as Copy Constructor and For Copying the values We have to pass the name of object whose values we wants to Copying and When we are using or passing an Object to a Constructor then we must have to use the & Ampersand or Address Operator.

Destructor: As we know that Constructor is that which is used for Assigning Some Values to data Members and For Assigning Some Values this May also used Some Memory so that to free up the Memory which is Allocated by Constructor, destructor is used which gets Automatically Called at the End of Program and we doesn’t have

to Explicitly Call a Destructor and Destructor Can't be Parameterized or a Copy This can be only one Means Default Destructor which Have no Arguments. For Declaring a Destructor we have to use ~tiled Symbol in front of Destructor

Access Specifiers

Access specifiers in C++ class defines the access control rules. C++ has 3 new keywords introduced, namely,

1.public

2.private

3.protected

These access specifiers are used to set boundaries for availability of members of class be it data members or member functions

Access specifiers in the program, are followed by a colon. You can use either one, two or all 3 specifiers in the same class to set different boundaries for different class members. They change the boundary for all the declarations that follow them.

this pointer: C++ uses a unique keyword called this to represent an object that invokes a member function. this is a 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.

Important notes on this pointer:

- this pointer stores the address of the class instance, to enable pointer access of the members to the member functions of the class.
- this pointer is not counted for calculating the size of the object.
- this pointers are not accessible for static member functions.
- this pointers are not modifiable.

Algorithm:

1. Design a Class Info which holds the appropriate data members to store students data.
2. Create a default constructor info::info() to initialize the data members.

3. Create a parameterized constructor to update the values into objects info::info(info &p)
4. Create functions to accept values from the user void info::accept()
5. Design the program to accept, insert and search and display a record based on users choice by using switch-case();
6. Create a function to display the student information void info::display()

Sample Code

```
#include<iostream>
#include<string.h>
using namespace std;
class info
{
  public:
    char dob[10];
    char add[30];
    char name[30],year[3];
    float marks;
    long tel_no;
    static int cnt;
    info();
    info(info &);
    void accept();
    void display();
    static int count();
};

info::info()// default constructor
{
    strcpy(name,"Default");
    strcpy(year,"FE");
    strcpy(add,"Default");
    marks=0;
    tel_no=10;
    strcpy(dob,"1/1/90");

}
info::info(info &p)//Parametrized constructor
```

```
{
   strcpy(name,p.name);
   strcpy(year,p.year);
   marks=p.marks;
   tel_no=p.tel_no;
   strcpy(dob,p.dob);

}
void info::accept()
{
   cout<<"Enter the name::";
   cin>>name;
   cout<<"Enter the address:";
   cin>>add;
   cout<<"Enter the year:";
   cin>>year;
   cout<<"Enter telephone number:";
   cin>>tel_no;
   cout<<"Enter Marks:";
   cin>>marks;
   cout<<"Enter date of birth:";
   cin>>dob;
}
void info::display()
{
   cout<<"\n\nName:"<<this->name;// this pointer
   cout<<"\nAddress:"<<this->add;
   cout<<"\nyear:"<<this->year;
   cout<<"\nTelephone number:"<<this->tel_no;
   cout<<"\nMarks:"<<this->marks;
   cout<<"\nDate of birth:"<<this->dob;
}
int info::cnt=0;
int info::count()
{
   cnt=cnt+1;
   return(cnt);
}
int main()
{
```

```
int ch,i,n,ch1,pos,temp,flag=0;
info o[50];
char name[12];
do
{
cout<<"\n\n==============" ;
cout<<"\n    MENU";
cout<<"\n==============";
cout<<"\n1.Accept";
cout<<"\n2.Display";
cout<<"\n3.Insert record";
cout<<"\n4.Search record";
cout<<"\n5.Delete record";
cout<<"\n6.Exit";
cout<<"\nEnter your choice:";
cin>>ch;
switch(ch)
{
case 1:cout<<"\n1.Inbuilt record..?";
       cout<<"\n2.New record..?";
       cin>>ch1;
       switch(ch1)
       {
        case 1:
              info a1(info o1);
              cout<<"\nDefault values initialized....";
              n=info::count();
               break;
        case 2:
               o[n].accept();//array of objects
               n=info::count();

               break;
       }


        break;
case 2:for(i=0;i<n;i++)
        {
               o[i].display();
```

```
        }
        break;
case 3:cout<<"\nEnter the position to insert record:";
        cin>>pos;

        n=info::count();
        pos=pos-1;
        for(i=n;i>=pos;i--)
        {
           temp=pos;
           o[i+1]=o[i];

        }

        o[temp].accept();
        break;
case 4:cout<<"Enter the name to be searched:";
        cin>>name;
        for(i=0;i<n;i++)
        {
           if(strcmp(name,o[i].name)==0)
           {
              flag=1;
              o[i].display();

              break;
           }
        }
        if(flag==0)
              cout<<"Sorry.. Record not found";
        break;

case 5:cout<<"\nEnter the position to delete record:";
        cin>>pos;
        pos=pos-1;
        for(i=pos;i<n;i++)
                {
                   temp=pos;
                   o[i]=o[i+1];
```

```
                }
        cout<<"\nRecord is deleted..";
        n=info::cnt-1;
        break;
    case 6:
        break;



   }
   }while(ch!=6);
   return 0;
}
```

Sample Output

```
==============
   MENU
==============
1.Accept
2.Display
3.Insert record
4.Search record
5.Delete record
6.Exit
Enter your choice:1

1.Inbuilt record..?
2.New record..?2
Enter the name::bob
Enter the address:pune
Enter the year:SE
Enter telephone number:123456
Enter Marks:65
Enter date of birth:2/2/22
```