**Practice the following programs on the compiler. These programs are based on Classes, Structures and Pointers. These programs in will help in learning the concepts Pointers and Linked List to be covered this week in the class.**

**There are no errors in these programs however if you find any bug then please try to fix it up. Based on these program we will have new programming assignment in next week or 3rd week of this academic session.**

**In case you do not understand the code then feel free to either ask the help of TAs or Me.**

**C++ Program (Code Snippet)–Accessing public data member (Variable) inside main function using Object Name**

**Let’s consider the following example:**

/\*C++ program to access public data member inside main using object name\*/

#include <iostream>

**using** **namespace** std;

**class** Sample{

**public**:

**int** x;

};

**int** main(){

Sample objSample;

objSample.x=20;

cout<<"value of x: "<<objSample.x<<endl;

**return** 0;

}

**Output**

value of x: 20

**C++ Program (Code Snippet)–Declare, Define and Access a Public Static Data Member inside main()**

**Let’s consider the following example:**

/\*C++ program to declare, define and access public static data member.\*/

#include <iostream>

**using** **namespace** std;

**class** Sample{

**public**:

//declaration of static data member

**static** **int** x;

};

//defining static data member of class

**int** Sample::x=0;

**int** main(){

//accessing static data member

cout<<"value of x: "<<Sample::x<<endl;

//modify the value of x

Sample::x=100;

cout<<"Modified value of x: "<<Sample::x<<endl;

**return** 0;

}

**Output**

value of x: 0

Modified value of x: 100

## C++ Program-Definitions of Constructor and Destructor inside class definition

**Let’s consider the following example:**

#include <iostream>

**using** **namespace** std;

**class** Example{

**public**:

//default constructor

Example(){cout<<"Constructor called."<<endl;}

//function to print message

**void** display(){

cout<<"display function called."<<endl;

}

//Destructor

~Example(){cout<<"Destructor called."<<endl;}

};

**int** main(){

//object creation

Example objE;

objE.display();

**return** 0;

}

### Output

Constructor called.

display function called.

Destructorcalled.

## C++ Program-Read and Print House details along with Room details

**Let’s consider the following example:**

/\*C++ Class Exercise - Read and Print House details along with Room details\*/

#include<iostream>

**using** **namespace** std;

**class** room {

**int** l;

**int** b;

**int** h;

**public** :

**void** getroom()

{

cout<<"Enter length, breath, height: ";

cin>>l>>b>>h;

}

**void** putroom()

{

cout<<"Length: "<<l<<",Breath: "<<b<<", Height: "<<h<<endl;

}

};

**class** address {

**int** hno;

**char** cty[30];

**char** state[30];

**public** :

**void** getad()

{

cout<<"house number : ";

cin>>hno;

cout<<"city :";

cin>>cty;

cout<<"state : ";

cin>>state;

}

**void** putad()

{

cout<<"House No.: "<<hno<<",city: "<<cty<<",state: "<<state<<endl;

}

};

**class** house{

**char** housename[30];

address a;

room r[10]; //max. 10 rooms

**public** :

**void** input();

**void** display();

};

//function definition

**void** house :: input()

{

cout<<"Enter house name: ";

cin>>housename;

cout<<"Enter Address : \n";

a.getad();

**for**(**int** i=0;i<3;i++){

cout<<"House Details : "<<i+1<<"\n";

r[i].getroom();

}

}

//function definition

**void** house :: display()

{

cout<<"House name: "<<housename<<endl;

cout<<"Address is: ";

**for**(**int** i=0;i<3;i++){

cout<<"House Details : "<<i+1<<"\n";

r[i].putroom();

}

}

**int** main()

{

house x;

x.input();

x.display();

**return** 0;

}

### Output

Enter house name: My\_Sweet\_Home

Enter Address :

house number : 101

city :NY

state : NY

House Details : 1

Enter length, breath, height: 10 10 10

House Details : 2

Enter length, breath, height: 10 10 8

House Details : 3

Enter length, breath, height: 20 20 10

House name: My\_Sweet\_Home

Address is: House Details : 1

Length: 10,Breath: 10, Height: 10

House Details : 2

Length: 10,Breath: 10, Height: 8

House Details : 3

Length: 20,Breath: 20, Height: 10

## C++ Program (Code Snippet)–Demonstrate use/access of Protected Data Member inside Derived class’s Public Member Function using C++ Inheritance

**Let’s consider the following example:**

/\*C++ program to demonstrate use of protected data members in inheritance\*/

#include <iostream>

**using** **namespace** std;

//class definition

**class** A{

**private**:

**int** a;

**protected**:

**int** p;

**public**:

**void** get\_a(**int** a){

**this**->a=a;

}

**void** put\_a(){

cout<<"a="<<a<<endl;

}

};

**class** B: **public** A{

**private**:

**int** b;

**public**:

**void** get\_b(**int** b){

**this**->b=b;

}

**void** get\_p(**int** p){

**this**->p=p;

}

**void** put\_b(){

cout<<"b="<<b<<endl;

}

**void** put\_p(){

cout<<"p="<<p<<endl;

}

};

**int** main(){

//creating object of B (derieved class)

B objB;

//get values of a,b and p

objB.get\_a(10);

objB.get\_b(20);

objB.get\_p(30);

//print values of a,b and p

objB.put\_a();

objB.put\_b();

objB.put\_p();

**return** 0;

}

### Output

a=10

b=20

p=30

## C++ Program (Code Snippet)–Access private member functions inside public member function in C++ class

**Let’s consider the following example:**

/\*C++ program to demonstrate calling of private member functions inside public member function\*/

#include <iostream>

**using** **namespace** std;

//class definition

**class** A{

**private**:

**int** a;

**int** b;

//set value of a

**void** set\_a(**int** a){

**this**->a=a;

}

//set value of b

**void** set\_b(**int** b){

**this**->b=b;

}

**public**:

**void** getValues(**int** x,**int** y){

set\_a(x); //calling private member function

set\_b(y); //calling private member function

}

**void** putValues(){

//printing values of private data members a,b

cout<<"a="<<a<<",b="<<b<<endl;

}

};

**int** main(){

//creating object

A objA;

//set values to class data members

objA.getValues(100,200);

//print values

objA.putValues();

**return** 0;

}

### Output

a=100,b=200

**C++ Program-Read and print Class, Student details using Two Classes**

**Let’s consider the following example:**

/\*C++ Class Exercise - Read and Print Class, Students

Details using Two Classes\*/

#include <iostream>

#include <string.h>

**using** **namespace** std;

**class** student{

**private**:

**char** name[30];

**int** rollNo;

**public**:

**void** getStudent(){

strcpy(name,"PIYA KAUSHAL");

rollNo=101;

}

**void** printStudent(){

cout<<"Name: "<<name<<",Roll No.: "<<rollNo<<endl;

}

};

**class** classDetails{

**private**:

**char** clsName[30];

student std; //object

**public**:

**void** getClassDetails(){

strcpy(clsName,"Higher Sec.");

std.getStudent();

}

**void** printClassDetails(){

cout<<"Class Name: "<<clsName<<endl;

std.printStudent();

}

};

**int** main()

{

classDetails CD;

CD.getClassDetails();

CD.printClassDetails();

**return** 0;

}

### Output

Class Name: Higher Sec.

Name: PIYA KAUSHAL,Roll No.: 101

## C++ Program-Demonstrate Example of Array of Structures

/\*C++ - program for Array of Structure.\*/

#include <iostream>

#include <iomanip>

**using** namespace std;

#define MAX 100

**struct** student{

**char** name[30];

**int** rollNumber;

};

**int** main(){

**struct** student std[MAX];

**int** n,loop;

cout<<"Enter total number of students: ";

cin>>n;

**for**(loop=0; loop<n; loop++){

cout<<"Enter name:";

cin.ignore(1);

cin.getline(std[loop].name,30);

cout<<"Enter roll number:";

cin>>std[loop].rollNumber;

}

cout<<"Entered records are:"<<endl;

cout<<setw(30)<<"Name"<<setw(20)<<"Roll Number"<<endl;

**for**(loop=0; loop<n; loop++){

cout<<setw(30)<<std[loop].name<<setw(10)<<std[loop].rollNumber<<endl;

}

**return** 0;

}

Enter total number of students: 2

Enter name:Mike

Enter roll number:101

Enter name:Monty

Enter roll number:102

Entered records are:

Name Roll Number

Mike 101

Monty 102

## C++ Program-Demonstrate Example of Nested Structure

/\*C++ - program for Nested Structure (Structure with in Structure).\*/

#include <iostream>

**using** namespace std;

**struct** date\_of\_birth{

**int** dd,mm,yy;

};

**struct** student{

**char** name[30];

**int** rollNumber;

date\_of\_birth dob;

};

**int** main(){

student s;

cout<<"Enter name : ";

cin.getline(s.name,25);

cout<<"Enter roll number : ";

cin>>s.rollNumber;

cout<<"Enter date of birth (dd mm yy) : " ;

cin>>s.dob.dd>>s.dob.mm>>s.dob.yy;

cout<<"Name:"<<s.name<<",Roll Number:"<<s.rollNumber<<endl;

cout<<"Date of birth:"<<s.dob.dd<<"/"<<s.dob.mm<<"/"<<s.dob.yy<<endl;

**return** 0;

}

Enter name : Mike

Enter roll number : 101

Enter date of birth (dd mm yy) : 29 09 2000

Name:Mike,Roll Number:101

Date of birth:29/9/2000

In this program we are reading Name, Roll number and Date of birth of a student, here Name and Roll Number are declared in **student structure**, while **date\_of\_birth** is another structure which contains **dd,mm,yy** variables to read date of birth. And we used **date\_of\_birth structure** in **student structure**.

# **C++ Program-Declare Integer variable dynamically, print the memory addresses**

Here, we will learn **how we can declare an integer variable dynamically and how to print address of declared memory block?**

**Consider the following program:**

#include <iostream>

**using** **namespace** std;

**int** main()

{

**int** a;

**int** \*ptr;

ptr=**new** **int**;

cout<<(&ptr)<<","<<ptr<<endl;

ptr=**new** **int**;

cout<<(&ptr)<<","<<ptr<<endl;

**delete** (ptr);

**return** 0;

}

# **C++ Program to declare read and print dynamic integer array**

**Here, we will learn how to declare, read and print dynamically allocated array? A dynamic array declares by new operator and deletes by delete operator. Here we will read total number of elements, read total elements and print them.**

**Consider the following program:**

#include <iostream>

**using** **namespace** std;

**int** main()

{

**int** \*arr;

**int** i,n;

cout<<"Enter total number of elements:";

cin>>n;

//declare memory

arr=**new** **int**(n);

cout<<"Input "<<n<<" elements"<<endl;

**for**(i=0;i<n;i++)

{

cout<<"Input element "<<i+1<<": ";

cin>>arr[i];

}

cout<<"Entered elements are: ";

**for**(i=0;i<n;i++)

{

cout<<arr[i]<<" ";

}

cout<<endl;

**delete** (arr);

**return** 0;

}