LECTURE-6

Nesting of member function

A member function can be called by using its name inside another member function of the same class. This is known as nesting of member functions. Nesting member functions can be a useful way to organize and encapsulate functionality within a class.

```
#include <iostream>
using namespace std;
class set
{
       int m,n;
       public:
              void input();
              void display ();
              int largest();
};
int set::largest ()
{
       if(m>n)
              return m;
       else
              return n;
}
void set::input()
{
       cout<<"input values of m:";
       cin>>m;
       cout<<"input values of n:";
       cin>>n;
}
void set::display()
{
       cout<<"largestvalue="<<largest()<<"\n";
}
main()
{
       set A;
       A.input();
       A.display();
}
```

```
#include<iostream>
using namespace std;
class val
{
       int x;
       public:
              void p()
              {
                     cin>>x;
              }
              void print()
              {
                     cout<<x;
              }
};
main()
{
       val obj;
       obj.p();
       obj.print();
}
```

```
#include<iostream>
using namespace std;
class nest
{
       int x,y;
       int sum()
       {
              return x+y;
       }
       public:
              int print()
              {
                     cout<<"Enter x and y: ";
                     cin>>x>>y;
                     cout<<"x+y= "<< sum();
              }
};
main()
{
       nest n;
       n.print();
}
```

```
#include<iostream>
using namespace std;
class nest
{
      int a;
      int square_num( )
      {
           return a*a;
      }
      public:
      void input_num( )
      {
      cout<<"\nEnter a number: ";
      cin>>a;
      }
      int cube_num( )
      {
           return a* a*a;
      }
       void disp_num()
       {
             int sq=square_num();//nesting of member function
             int cu=cube_num();  //nesting of member function
             cout<<"\nThe square of: "<<a<<" is  " <<sq;
             cout<<"\nThe cube of: "<<a<<" is "<<cu;
       }
};

main()
{
      nest n1;
      n1.input_num();
      n1.disp_num();
}
```

If there are multiple variables with the same name defined in separate blocks then :: (scope resolution) operator will reveal the hidden file scope(global) variable.

```
#include<iostream>
using namespace std;
int a=100;
class A

{
    int a;
    public:
```

```
              void fun()
             {
             a=20;
             a=a+::a;          //using global variable value
             cout<<a;
             }
};
A a1;
main()
{
     a1.fun();
}
```

```
#include <iostream>
using namespace std;
class average
{
       int a,b;
       public:
              void read();
              void print();
              int avg();
};
void average::read()
{
       cout<<"\n enter a and b: ";
       cin>>a>>b;
}
void average::print()
{
       cout<<"value of a: "<<a;
       cout<<"\nvalue of b: "<<b;
       cout<<"\naverage is : "<<avg();
}
int average::avg()
{
       return (a+b)/2;
}
main()
{
       average A;
       A.read();
       A.print();
}
```

Private member functions:

Although it is a normal practice to place all the data items in a private section and all the functions in public, some situations may require contain functions to be hidden from the outside calls. Tasks such as deleting an account in a customer file or providing increment to and employee are events of serious consequences and therefore the functions handling such tasks should have restricted access. We can place these functions in the private section.

A private member function can only be called by another function that is a member of its class. Even an object cannot invoke a private function using the dot operator.

```
Class sample
{
        int m;
        void read (void);
        void write (void);
};
```

if **s** is an object of sample, then s.read(); is illegal. How ever the function read() can be called by the function update ( ) to update the value of m.

```
void sample :: update(void)
        {
              read( );
        }
```

```
Class sample
{
int m;
    void read (void);
public:
    void write (void)
    {
        read();
    }
};
```

If we create an object s1 of class sample, s1.read() or s1.write() is illegal. Because object cannot access the private data outside of the class. One solution to access the private member function is, to call private member function from the member function declared inside the public section of the class.

```
#include<iostream>
using namespace std;
class part
{
        private:
                int   modelnum,partnum;
                float cost;
        public:
                void setpart ( int mn, int pn ,float c)
                {
                        modelnum=mn;
                        partnum=pn;
                        cost=c;
                }
                void showpart ( )
                {
                cout<<"model:"<<modelnum<<endl;
                cout<<"num:"<< partnum <<endl;
                cout<<"cost:"<<"$"<<cost;
                }
};
main()
{
        part  p1,p2;
        p1.setpart(64,73,217.55);
        p2.setpart(57,89,789.55);
        p1.showpart();
        p2.showpart();
}
```

```
#include<iostream>
using namespace std;
class dist
{
	int feet;
	float inches;
	public:
		void setdist (int ft, float in)
		{
			feet=ft;
			inches=in;
		}
		void getdist()
		{
			cout<<"enter feet:";
			cin>>feet;
			cout<<"enter  inches:";
			cin>>inches;
		}
		void showdist()
		{
			cout<< feet<<"_"<<inches<<endl;
		}
};
main()
{
	dist d1;
	d1.setdist(11,6.25);
	cout<<endl<<"dist:";
	d1.showdist();
}
```

\

Q: Write C++ program to demonstrate the private member function:

```
#include <iostream>
using namespace std;
class Student {
private:
    int rNo;
    float perc;
    //private member functions
    void inputOn(void)
    {
        cout << "Input start..." << endl;
    }
    void inputOff(void)
    {
        cout << "Input end..." << endl;
    }
public:
    //public member functions
    void read(void)
    {
        //calling first member function
        inputOn();
        //read rNo and perc
        cout << "Enter roll number: ";
        cin >> rNo;
        cout << "Enter percentage: ";
        cin >> perc;
        //calling second member function
        inputOff();
    }
    void print(void)
    {
        cout << endl;
        cout << "Roll Number: " << rNo << endl;
        cout << "Percentage: " << perc << "%" << endl;
    }
};
main()
{
    Student std;
    std.read();
    std.print();
}
```