Scope

- Member initializer
- Composition – object as members of other classes

Member initializer

- Constructor definition uses a member initializer list to initialize class data members
- Appear between a constructor's parameter list and the left brace that begins the constructor's body
- Separated by colon :

Example

Member initializer

```
class time
{
public:
int hour; int minute; int
   seconds;
};
```

```
time::time(int i,int j,int k) :
hour(i), minute(j),
   seconds(k){ }
```

```
void main()
{
time noon(12,0,0);
noon.minute=22;
}
```

Contd..

- Member initializer list executes before the body of the constructor executes

- Const data members and data members that are references must be initialized using member initializers

Composition - Objects as members of classes

- A capability to use object as a member of another class is composition

- How an object's constructor can pass arguments to member-object constructors?

Example

```
#include<iostream.h>
#include<conio.h>
class member
{
  private:
  int class1member;
  public:
  member() { cout<<"Calling
    member default
    constructor"<<endl;
    class1member=10; }
  member(int i) : class1member(i) {
    cout<<"Calling member
    parameterized cons"<<endl;}
  ~member() { cout<<" Calling
    member destructor "<<endl; }
```

```
class container
{
  private:
  member a;
  member b;
  public:
  container() { cout<<"Calling container
default constructor "<<endl;}
  container(int i) : b(i) { cout<<" Calling
container para cons"<< endl;}
  ~container() { cout<<" Calling container
destructor "<<endl; }
};
```

```
void main()
{
  clrscr();
//  container obj1;
  container obj2(20);
  getch();
}
```

composition

- Member objects are constructed in the order in which they are declared in the class definition and before their enclosing class objects are constructed

- Destructors are called in reverse order

Programming assignments

- Write a class for time having hours, minutes and seconds as data members. Write a member function calculates the amount of time in seconds between two times.

Copy constructor

- Initializes an array by making a copy of an existing object
- It is takes a reference to an object of the same class as itself as an argument

Example

```
#include<iostream>
class code
{       private: int id;
        public:
                code(int a) { id=a;}
                code(code & x) { id=x.id; }
                void display(){ std::cout<<id ; }
 }
void main()
{ code a(100);
code b(a); // copy constructor called
a.display(); b.display();
}
```

Assignment

- Explain copy constructor?