CONST MEMBER FUNCTION

If a member function does not alter any data in the class, then we may declare it as a const member function –

```
void mul(int, int) const;
double getBalance() const;
```

The qualifier const is appended to the function prototypes in both declaration and definition.`

Compiler will generate error if such function try to alter the data value

POINTER TO MEMBERS

- It is possible to take the address of a member of a class and assign it to a pointer
- The address of a member can be obtained by applying the operator & to a “fully qualified” class member name.
- A class member pointer can be declared using the operator :: * with the class name.

```
class A
{
    private:
        int m;
    public:
        void show( );
};
```

We can define a pointer to the member m as follows :

```
int A :: * ip = & A :: m;
```

LOCAL CLASS

- A class declared inside a function becomes local to that function and is called Local Class in C++.
- A local class name can only be used locally i.e., inside the function and not outside it.
- The methods of a local class must be defined inside it only.
- A local class can have static functions but, not static data members.

FACTS ABOUT LOCAL CLASS

1) A local class type name can only be used in the enclosing function.

2) **All the methods of Local classes must be defined inside the class only.**

3) A Local class cannot contain static data members. It may contain static functions though.

4) Member methods of the local class can only access static and enum variables of the enclosing function. Non-static variables of the enclosing function are not accessible inside local classes

5) Local classes can access global types, variables, and functions.

CONSTRUCTOR

- A constructor is a special member function whose task is to initialize the objects of its class .
- It is special because its name is the same as the class name.
- The constructor is invoked when ever an object of its associated class is created.
- It is called constructor
- because it construct the values of data members of the class.

```
//class with a constructor
class integer
{
    int m,n;
    public:
        integer(void);//constructor declared
};
integer :: integer()
{
    m = 0;
    n = 0;
}
```

When a class contains a constructor like the one defined above it is guaranteed that an object created by the class will be initialized automatically.

For example:- **Integer int1;** //object int1 created

This declaration not only creates the object int1 of type integer but also initializes its data members m and n to zero

CHARACTERISTICS OF CONSTRUCTOR

- They should be declared in the public section .
- They are invoked automatically when the objects are created.
- They don't have return types, not even void and therefore they cannot return values.
- They cannot be inherited , though a derived class can call the base class constructor .
- Like other C++ function , they can have default arguments,
- Constructor can't be virtual.
- An object with a constructor can't be used as a member of union.

TYPES OF CONSTRUCTOR

- Default
  - Class_name();
- Parameterized
  - Class_name(parameters);
- Copy
  - Class_name(const Class_name oldObject)

DEFAULT CONSTRUCTOR

- A constructor that accept no parameter is called the default constructor.
- The default constructor for class A is A :: A( ).
- If no such constructor is defined, then the compiler supplies a default constructor .
- Therefore a statement such as :- A a ;//invokes the default constructor of the compiler of the compiler to create the object "a"

PARAMETERIZED CONSTRUCTOR

1. It is used to initialize the various data elements of different objects with different values when they are created.
2. It is used to overload constructors.
3. We can have more than one constructor in a class, it is called as Constructor Overloading

COPY CONSTRUCTOR

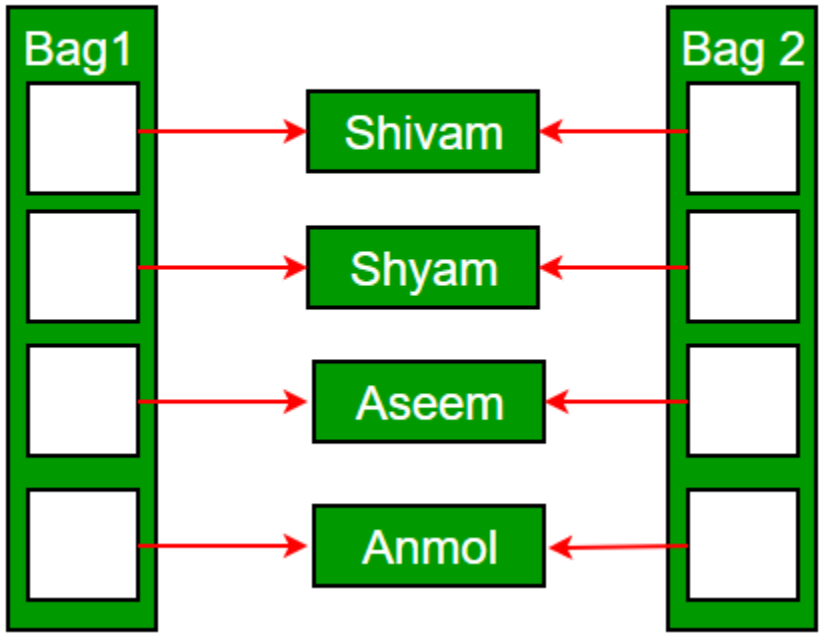
- A copy constructor is a member function that initializes an object using another object of the same class. A copy constructor has the following general function prototype:

```
ClassName (const ClassName &old_obj);
```

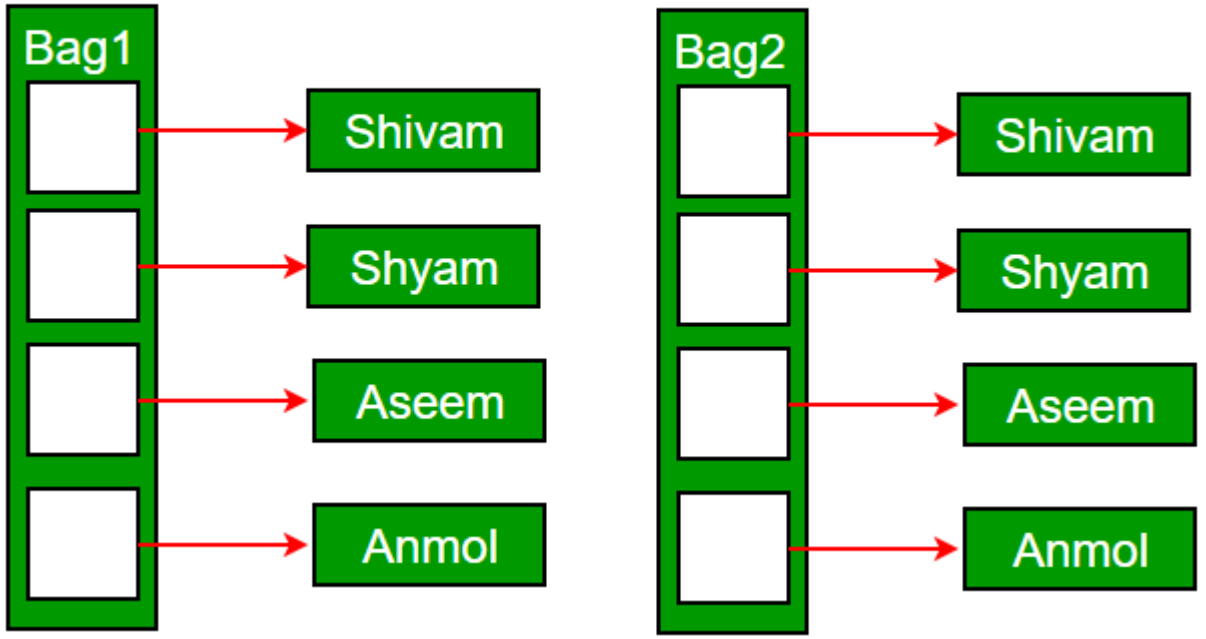
When is copy constructor called -

1. When an object of the class is returned by value.
2. When an object of the class is passed (to a function) by value as an argument.
3. When an object is constructed based on another object of the same class.

Shallow Copy



Deep Copy



UNION

- unions are the same as the structure type, means a collection of similar and different data types.
- But where the memory location of the members of the structure is different for each member, the union reserves same location in the memory to all its data member, due to which they affect each other,
- When you enter the second value after entering the first value, the second value erase the first value, in this way only the value entered in the last is stored, and when we print these values, in the place of erasing values, we get garbage values, that is only the value entered in last in print.
- In the diagram below, you can see that the union reserves the same location in the memory for its data members
- That is, the memory address of all the members is the same.