STATIC FUNCTION-ASSIGNMENT AND COPY INITIALIZATION

STATIC DATA MEMBER:

When we define the data member of a class using the static keyword, the data members are called the static data member. A static data member is similar to the static member function because the static data can only be accessed using the static data member or static member function. And, all the objects of the class share the same copy of the static member to access the static data.

Syntax

```
static data_type data_member;
```

```
#include <iostream>
#include <string.h>
using namespace std;
// create class of the Car
class Car
{
private:
int car_id;
char car_name[20];
int marks;

public:
// declare a static data member
static int static_member;

Car()
{
static_member++;
}

void inp()
{
cout << " \n\n Enter the Id of the Car: " << endl;
cin >> car_id; // input the id
cout << " Enter the name of the Car: " << endl;
cin  >> car_name;
cout << " Number of the Marks (1 - 10): " << endl;
cin >> marks;
}

// display the entered details
void disp ()
{
cout << " \n Id of the Car: " << car_id;
cout << "\n Name of the Car: " << car_name;
cout << " \n Marks: " << marks;

}  };
// initialized the static data member to 0
```

```
int Car::static_member = 0;
 int main ()
{  // create object for the class Car
Car c1;
// call inp() function to insert values
c1. inp ();
c1. disp();  //create another object
Car c2;
// call inp() function to insert values
c2. inp ();
c2. disp();
 cout << " \n No. of objects created in the class: " << Car :: static_member <<endl;
return 0;
}
```

Output

```
Enter the Id of the Car:
101
Enter the name of the Car:
Ferrari
Number of the Marks (1 - 10):
10
Id of the Car: 101
Name of the Car: Ferrari
Marks: 10
Enter the Id of the Car:
205
Enter the name of the Car:
Mercedes
Number of the Marks (1 - 10):
9
Id of the Car: 205
Name of the Car: Mercedes
Marks: 9
No. of objects created in the class: 2
```

Copy Constructor Or Copy Initialization (Syntax)

```
classname (const classname &obj) {
  // body of constructor
}
```

Direct Initialization or Assignment Operator (Syntax)

```
classname Ob1, Ob2;
Ob2 = Ob1;
```

Copy initialization	Direct Initialization
The Copy initialization is basically an overloaded constructor	Direct initialization can be done using assignment operator.
This initializes the new object with an already existing object	This assigns the value of one object to another object both of which are already exists.
Copy initialization is used when a new object is created with some existing object	This is used when we want to assign existing object to new object.
Both the objects uses separate memory locations.	One memory location is used but different reference variables are pointing to the same location.
If no copy constructor is defined in the class, the compiler provides one.	If the assignment operator is not overloaded then bitwise copy will be made