Static Member Functions

Muhammad Hammad Waseem

m.hammad.wasim@gmail.com

Introduction

- In previous lecture we introduced static data members.
- As you may recall,
  - a static data member is not duplicated for each object; rather a single data item is shared by all objects of a class.
- Let’s extend this concept by showing how functions as well as data may be static.

Static Function

- A type of member function that can be accessed without any object of class is called **static function**.
- Normally, a member function of any class cannot be accessed or executed without creating an object of that class.
- In some situations, a member function has to be executed without referencing any object.
- The static data members of a class are not created for each object.
- The class creates only one data member for all objects.
- The static data member is defined when the program is executed.
- The program may require to access a static data member before creating an object.
- The static member functions can be used to access a static data member.

Accessing Static Functions

- When a data member is declared static, there is only one such data value for the entire class, no matter how many objects of the class are created.
- We shouldn’t need to refer to a specific object when we’re doing something that relates to the entire class.
- It’s more reasonable to use the name of the class itself with the scope-resolution operator.
- `class_name::static_FuncName();`
- However, this won’t work on a normal member function; an object and the dot member-access operator are required in such cases.
- To access function using only the class name, we must declare it to be a static member function.

Example 1

```
class Test
{
private:
static int n;
public:
static void show()
{
cout<<"n = "<<n<<endl;
}
};
```

```
int Test::n=10;
void main()
{
Test::show();
getch();
}
```

Output:
n = 10

How above Program Works

- The above program declares a class Test with a static data member n.
- The following statement defines the data member with an initial value of 10;
- `int Test::n=10;`
- The static data member exists in the memory even before creating any object.
- The program also declares a static member function show() that displays the value of a.
- The program calls the static member function without creating any objet of the class as follows:
- `Test::show();`

Example 2

```
class yahoo
{
private:
static int n;
public:
Yahoo()
{n++;}
static void show()
{
cout<<"you've created"<<n<<"
objects so far"<<endl;
}
};
int yahoo::n=0;
void main()
{
yahoo::show();
yahoo x,y;
x.show();
yahoo z;
x.show();
getch();
}
```

Output:

You've created 0 objects so far
You've created 2 objects so far
You've created 3 objects so far

How above Program Works

- The above program declares a static data member n to count the number of objects that have been created.
- The following statement defines variable: `int yahoo::n=0;`
- The above statement defines the variable and initializes it to 0.
- The above program creates three objects x, y and z. Each time an object is created the constructor is executed that increases the value of n by 1.
- The program also declares a static member function show() that displays the value of data member.
- It can be accessed directly by using class name followed by scope resolution operator :: and function name as: yahoo:: `show();`