Tutorial Link https://codequotient.com/tutorials/Static Class members/5b38c21fc6a1d0259e728e63

TUTORIAL

Static Class members

Chapter

1. Static Class members

Static Class members (Static data members & Static member functions)

A static class member has only one copy, belonging to the class instead of the instances. All instances share the same storage for a static class member. In C++ it is necessary to explicitly define static members. The Syntax is

```
static < variable definition>
static < function definition>
```

Examples are

```
static int c;
static void display ( )
{
        …….
}
```

A static member is referenced via scope resolution operator in the form of

```
ClassName :: VariableName or ClassName :: FunctionName( ).
```

- It can be used to implement "global" class variables and functions that can be used without creating instances of a class.
- It can also be used to share information among all instances, e.g., a count on the number of instances created.
- A static function can only access static variables, and cannot access non-static variables. A static function can be referenced without any instantiation (i.e., no instance is created). It can be invoked using the class name.

The following example displays the value of static data member.

C++

```
#include<iostream>
#include<cstdio>
#include<cmath>
// Include headers as needed

using namespace std;

class Number
{
  static int C;
  public:
  void count ( )
  {
    ++C;
    cout<<"\n C ="<< C;
  }
};

int Number :: C = 0;

int main( )
{
```

```
  Number a, b, c;
  a.count( );
  b.count( );
  c.count( );
  return 0;
}
```

Output:

```
C=1
C=2
C=3
```

The next example declares the static member functions and call them from the main ( ) function.

C++

```
#include<iostream>
#include<cstdio>
#include<cmath>
// Include headers as needed

using namespace std;

class Num
{
  static int C;
  public:
  static void count ( )
  {
    C++;
  }
  static void display( )
  {
    cout<<"\n value of C : "<< C;
  }
```

```
};

int Num :: C = 0;

int main( )
{
  Num :: display ( );
  Num :: count ( );
  Num :: count( );
  Num :: display ( );
}
```

Output:

```
Value of C : 0
Value of C : 2
```