Some interesting facts about static member functions in C++

Friend class and function in C++

Local Classes in C++

Nested Classes in C++

Simulating final class in C++

Constructors in C++

Copy Constructor in C++

Destructors in C++

Does C++ compiler create default constructor when we write our own?

When should we write our own copy constructor?

When is copy

constructor called?

Initialization of data members

Use of explicit keyword in C++

When do we use Initializer List in C++?

time.h header file in C with Examples

scanf("%[^\n]s", str) Vs gets(str) in C with Examples

C program to Insert an element in an Array

Types of Literals in C/C++ with Examples

Conditional or Ternary Operator (?:) in C/C++

Local Classes in C++

A class declared inside a function becomes local to that function and is called Local Class in C++. For example, in the following program, Test is a local class in fun().

```
#include<iostream>
using namespace std;

void fun()
{
      class Test  // local to fun
      {
        /* members of Test class */
      };
}

int main()
{
    return 0;
}
```

Following are some interesting facts about local classes.

1) *A local class type name can only be used in the enclosing function.* For example, in the following program, declarations of t and tp are valid in fun(), but invalid in main().

```
#include<iostream>
using namespace std;

void fun()
{
      // Local class
      class Test
      {
        /* ... */
      };

      Test t;  // Fine
      Test *tp;  // Fine
}

int main()
{
    Test t;  // Error
    Test *tp;  // Error
    return 0;
}
```

2) *All the methods of Local classes must be defined inside the class only.* For example, program 1 works fine and program 2 fails in compilation.

```
// PROGRAM 1
#include<iostream>
using namespace std;

void fun()
{
    class Test  // local to fun
    {
    public:

       // Fine as the method is defined inside the local class
       void method() {
          cout << "Local Class method() called";
       }
    };

    Test t;
    t.method();
}

int main()
{
    fun();
    return 0;
}
```

Output:

```
Local Class method() called
```

```
// PROGRAM 2
#include<iostream>
using namespace std;

void fun()
{
    class Test  // local to fun
    {
    public:
        void method();
    };

    // Error as the method is defined outside the local class
    void Test::method()
    {
        cout << "Local Class method()";
    }
}

int main()
{
    return 0;
}
```

Output:

```
Compiler Error:
 In function 'void fun()':
 error: a function-definition is not allowed here before '{' token
```

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

For example, program 1 fails in compilation, but program 2 works fine.

```
// PROGRAM 1
#include<iostream>
using namespace std;

void fun()
{
      class Test  // local to fun
      {
         static int i;
      };
}

int main()
{
    return 0;
}
```

```
Compiler Error:
 In function 'void fun()':
 error: local class 'class fun()::Test' shall not have static data member
```

```cpp
// PROGRAM 2
#include<iostream>
using namespace std;

void fun()
{
    class Test  // local to fun
    {
    public:
        static void method()
        {
            cout << "Local Class method() called";
        }
    };

    Test::method();
}

int main()
{
    fun();
    return 0;
}
```

Output:

```
Local Class method() called
```

4) *Member methods of 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.*

For example, the program 1 compiles and runs fine. But, program 2 fails in compilation.

```
// PROGRAM 1
#include<iostream>
using namespace std;

void fun()
{
      static int x;
      enum {i = 1, j = 2};

      // Local class
      class Test
      {
        public:
          void method() {
              cout << "x = " << x << endl;   // fine as x is static
              cout << "i = " << i << endl;   // fine as i is enum
          }
      };

      Test t;
      t.method();
}

int main()
{
    fun();
    return 0;
}
```

Output:

```
x = 0
i = 1
```

```
// PROGRAM 2
#include<iostream>
using namespace std;

void fun()
{
      int x;

      // Local class
      class Test
      {
        public:
          void method() {
              cout << "x = " << x << endl;
          }
      };

      Test t;
      t.method();
}

int main()
{
    fun();
    return 0;
}
```

Output:

```
In member function 'void fun()::Test::method()':
error: use of 'auto' variable from containing function
```

5) *Local classes can access global types, variables and functions. Also, local classes can access other local classes of same function.*. For example, following program works fine.

```
#include<iostream>
using namespace std;

int x;

void fun()
{

      // First Local class
      class Test1 {
      public:
         Test1() { cout << "Test1::Test1()" << endl; }
      };

      // Second Local class
      class Test2
      {
           // Fine: A local class can use other local classes of same
           Test1 t1;
      public:
          void method() {
              // Fine: Local class member methods can access global va
              cout << "x = " << x << endl;
          }
      };

      Test2 t;
      t.method();
}

int main()
{
    fun();
    return 0;
}
```

Output:

```
Test1::Test1()
x = 0
```

Also see Nested Classes in C++

References:

Local classes (C++ only)

Local Classes

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

Local Labels in C

How to return a local array from a C/C++ function?

Can we access global variable if there is a local variable with same name?

Storage Classes in C

Nested Classes in C++

Anonymous classes in C++

C++ Classes and Objects

Trivial classes in C++

C++ Stream Classes Structure

Storage Classes in C++ with Examples

File Handling through C++ Classes

Virtual functions in derived classes

C | Storage Classes and Type Qualifiers | Question 7

C | Storage Classes and Type Qualifiers | Question 6

C | Storage Classes and Type Qualifiers | Question 19

Article Tags : C C++

Practice Tags : C CPP

2

3.3

To-do Done

Based on **22** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org