Lecture 5

CSE202: OBJECT ORIENTED PROGRAMMING

Outline

- Inline member functions
- Non-inline member functions
- Static Data Members
- Static Member functions

C++ inline function

- C++ **inline** function is powerful concept that is commonly used with classes. If a function is inline, the compiler places a copy of the code of that function at each point where the function is called at compile time.
- Every time a function is called, it takes a lot of extra time in executing a series of instructions for tasks such as jumping to the function, saving registers, pushing arguments into the stack, and returning to the calling function.
- When a function is small, a substantial percentage of execution time may be spent in such overheads.
- To eliminate the cost of calls to smaller functions, C++ introduces a new feature called inline function, which is expanded in line when it is invoked.

Program example of Inline function

```
#include <iostream>
 using namespace std;
inline int Max(int x, int y) {
   return (x > y)? x : y;
}
int main() {
   cout << "Max (20,10): " << Max(20,10) << endl;
   cout << "Max (0,200): " << Max(0,200) << endl;
   cout << "Max (100,1010): " << Max(100,1010) << endl;
   return 0;
}
```

Inline Member Function

- To inline a function, place the keyword **inline** before the function name and define the function before any calls are made to the function. The compiler can ignore the inline qualifier in case defined function is more than 3-5 lines.
- A function definition in a class definition is an inline function definition, even without the use of the **inline** specifier.

A member function that is defined inside its class member list is called an **inline member function**.

An equivalent way to declare an inline member function is to either declare it in the class with the **inline** keyword (and define the function outside of its class) or to define it outside of the class declaration using the **inline** keyword.

Program example of Inline Member function

```
#include <iostream>
using namespace std;
class operation
{
        int a,b,add;
public:
void get() // inline Member Function
{
        cout << "Enter first value:";
        cin >> a;
        cout << "Enter second value:";
        cin >> b;
}
        void sum();
};s;
inline void operation :: sum()
{
        add = a+b;
cout <<"Addition of two numbers: " << a+b;
}
```

Method 1: Inline Member Function

```
int main()
{
cout << "Program using inline function\n";
s.get();
s.sum();
return 0;
}
```

Method 2: Inline Member Function

When not to use Inline Function

Note: Inlining is **only a request** to the compiler, not a command. **Compiler can ignore the request for inlining. Compiler may not perform inlining in such circumstances like:**

1) If a function contains a loop. (for, while, do-while)
2) If a function contains static variables.
3) If a function is recursive.
4) For Functions not returning values, if a return statement exists
5) If a function contains switch or goto statement.

Advantages and Disadvantages of Inline Function

Advantages of inline function

- Function call overhead doesn't occur.
- It also saves the overhead of push/pop variables on the stack when function is called.
- It also saves overhead of a return call from a function.
- After in-lining compiler can also apply intraprocedural optimization if specified
- Increases locality of reference by utilizing instruction cache

Disadvantages of inline function

- May increase function size so that it may not fit on the cache, causing lots of cache miss
- After in-lining function if variables number which are going to use register increases than they may create overhead on register variable resource utilization
- It may cause compilation overhead as if some body changes code inside inline function than all calling location will also be recompiled
- If used in header file, it will make your header file size large and may also make it unreadable.

MCQ

When the function is defined inside a class, it is treated as …….

A) data function
B) inline function
C) non inline function
D) member variable

Solution

When the function is defined inside a class, it is treated as …….

A) data function
B) **inline function**
C) non inline function
D) member variable

MCQ

Function which are declared in a class declaration and defined outside the class is known as…….

A) inline member function
B) non-inline member function
C) static member function
D) dynamic member function

Solution

Function which are declared in a class declaration and defined outside the class is known as…….

A) inline member function
B) **non-inline member function**
C) static member function
D) dynamic member function

MCQ

True about inline function statements in C++

```
class A
{ public:
 void func1()
{ }
 void func2();
};
 inline void A::func2()
{ }
```

1.Func1 is inline function
2.Func2 only is inline function
3.Func1 and Func2 both are inline functions
4.None of the above is inline

Solution

True about inline function statements in C++

```
class A
{ public:
 void func1()
{ }
 void func2();
};
 inline void A::func2()
{ }
```

1.Func1 is inline function
2.Func2 only is inline function
3.Func1 and Func2 both are inline functions
4.None of the above is inline

Non-Inline Member Function

- Non-Inline Member Function is basically same as Normal function in C++.
- It promotes code reuse and makes the program modular. During function calls, a lot of overhead tasks are performed like saving registers, pushing arguments to the stack, and returning to the calling function.

Syntax:

```
return_type_name function_name( parameters)
{
// Insert your Function code here
}
```

Program example of Non-Inline function

```
#include <iostream>
 using namespace std;
 int square(int s)
{
    return s * s;
}
int main()
{
        cout << "Enter number to compute its square : 5 " << endl;
     cout << "Square is : " << square(5) << endl;
   return 0;
}
```

MCQ

Which of the following statements about non-inline functions is true?

A)Non-inline functions must be defined in the same file where they are declared.

B)B) Non-inline functions are automatically inlined by the compiler.

C)C) Non-inline functions result in faster program execution compared to inline functions.

D)D) Non-inline functions can only be called from within the same class where they are defined.

Solution

Which of the following statements about non-inline functions is true?

A)Non-inline functions must be defined in the same file where they are declared.

B)B) Non-inline functions are automatically inlined by the compiler.

C)C) Non-inline functions result in faster program execution compared to inline functions.

D)D) Non-inline functions can only be called from within the same class where they are defined.

MCQ

How is a non-inline function declared in C++?

A)By using the external keyword.

B)By placing the noninline modifier before the function name.

C)By using the noinline keyword.

D)Non-inline functions don't need any special keyword for declaration.

Solution

How is a non-inline function declared in C++?

A)By using the external keyword.

B)By placing the noninline modifier before the function name.

C)By using the noinline keyword.

D)Non-inline functions don't need any special keyword for declaration.

Practice Questions

1. Write a program to find sum of 2 integers using inline function.
2. Write a program to find the multiplication values and the cubic values using the inline function.
3. Write a program for Inline Function without Class.

Static Data Members

Static data members are class members that are declared using **static** keywords. A static member has certain special characteristics which are as follows:

•Only one copy of that member is created for the entire class and is shared by all the objects of that class, no matter how many objects are created.

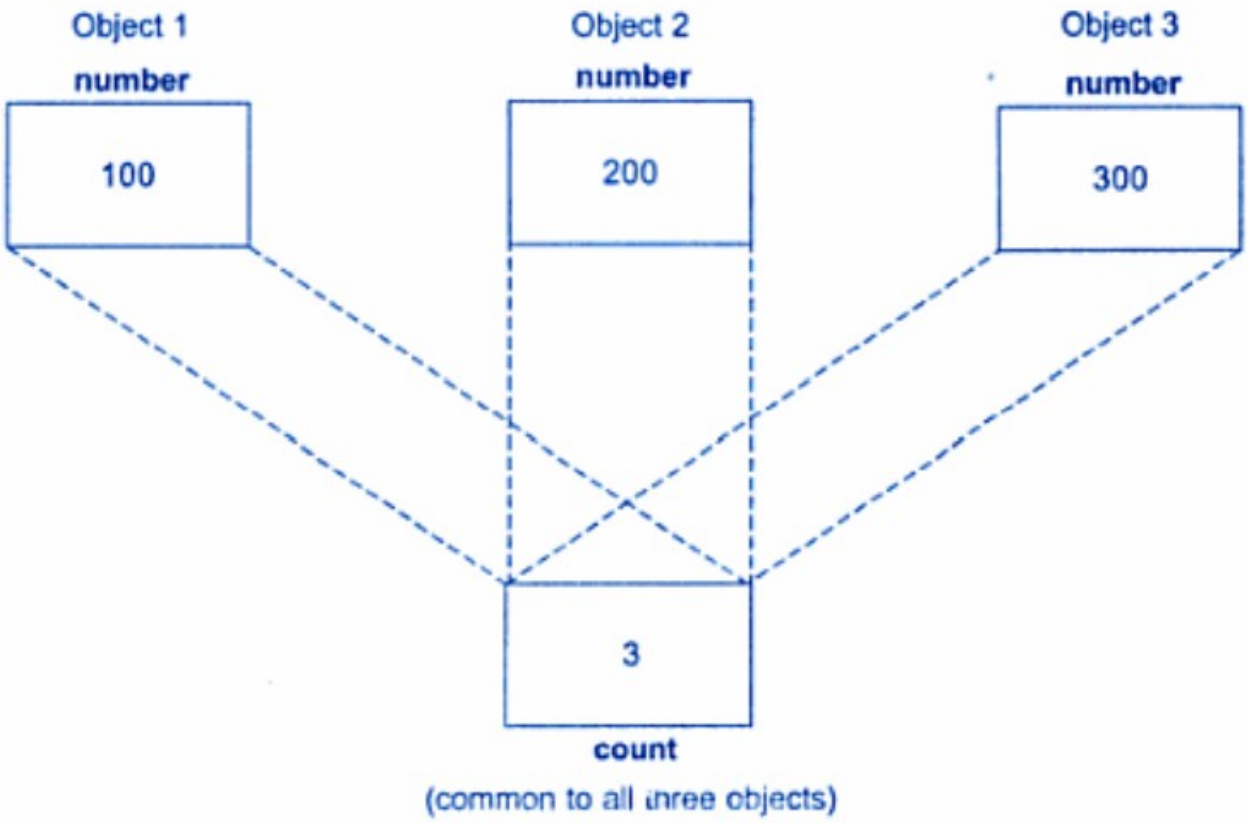
•It is initialized before any object of this class is created, even before the main starts.

•It is visible only within the class, but its lifetime is the entire program.

Syntax:

static data_type data_member_name;

Static data member(count)



Program example of static data member

```
#include <iostream>
#include<string.h>
 using namespace std;
class Student
{
 private:
int rollNo;
char name[10];
  int marks;
public:
static int objectCount;
Student()
{
objectCount++;
}
void getdata()
{
cout << "Enter roll number: "<<endl;
 cin >> rollNo;
cout << "Enter name: "<<endl;
```

Continued..

```
cin >> name;
 cout << "Enter marks: "<<endl;
cin >> marks;
}
void putdata()
{       cout<<"Roll Number = "<<
rollNo <<endl;
cout<<"Name = "<< name <<endl;
cout<<"Marks = "<< marks <<endl;
 cout<<endl;
 }
};
int Student::objectCount = 0;
 int main(void)
{
Student s1;
s1.getdata();
s1.putdata();
Student s2;
```

Continued..

```
s2.getdata();

s2.putdata();

Students3;

s3.getdata();

 s3.putdata();

 cout << "Total objects created = " <<
Student::objectCount << endl;

return 0;

}
```

Static Member functions

Static Member Function in a class is the function that is declared as static because of which function attains certain properties as defined below:

- A static member function is independent of any object of the class.
- A static member function can be called even if no objects of the class exist.
- A static member function can also be accessed using the class name through the scope resolution operator.
- A static member function can access static data members and static member functions inside or outside of the class.
- Static member functions have a scope inside the class and cannot access the current object pointer.
- You can also use a static member function to determine how many objects of the class have been created.

MCQ

Which of the following is true about static member functions in C++?

a)They have access to all non-static members of the class.

b)They can be called using the dot (.) operator.

c)They are automatically called when an instance of the class is created.

d)They don't have a 'this' pointer.

Solution

Which of the following is true about static member functions in C++?

a)They have access to all non-static members of the class.

b)They can be called using the dot (.) operator.

c)They are automatically called when an instance of the class is created.

d)They don't have a 'this' pointer.

MCQ

What is the benefit of using a static member function in C++?

a)It can modify all instances' data members simultaneously.

b)b) It can access private data members of the class.

c) It can be called without creating an instance of the class.

d) d) It can be overridden in derived classes.

Solution

What is the benefit of using a static member function in C++?

a)It can modify all instances' data members simultaneously.

b) It can access private data members of the class.

c) It can be called without creating an instance of the class.

d) It can be overridden in derived classes.

Need of Static Member functions

- Static members are frequently used to store information that is shared by all objects in a class.
- For instance, you may keep track of the quantity of newly generated objects of a specific class type using a static data member as a counter. This static data member can be increased each time an object is generated to keep track of the overall number of objects.

Program example of Static Member functions

```
include <iostream>
using namespace std;
 class Box
{
 private:
    static int length;
    static int breadth;
    static int height;
     public:
static void print()
    {
        cout << "The value of the length is: " << length << endl;
        cout << "The value of the breadth is: " << breadth << endl;
        cout << "The value of the height is: " << height << endl;
    }
};
```

```
int Box :: length = 10;

int Box :: breadth = 20;

int Box :: height = 30;

 int main()

{

    Box b;

cout << "Static member function is called through Object name: \n" << endl;

   b.print();

cout << "\nStatic member function is called through Class name: \n" << endl;

    Box::print();

      return 0;

}
```

MCQ

What is a static member function in C++?

a)A member function that can only be called from objects of the class.

b)b) A member function that is declared as 'const' within the class.

c)c) A member function that belongs to the class rather than to any specific object of the class.

d)d) A member function that has access to private data members of the class.

Solution

What is a static member function in C++?

a)A member function that can only be called from objects of the class.

b)A member function that is declared as 'const' within the class.

c) A member function that belongs to the class rather than to any specific object of the class.

d)A member function that has access to private data members of the class.

MCQ

Which of the following is true about static member functions in C++?

a) They can access non-static data members directly.

b) They can be called using the dot (.) operator.

c) They have a 'this' pointer pointing to the current object.

d) They are invoked using the class name without creating an instance of the class.

Solution

Which of the following is true about static member functions in C++?

a) They can access non-static data members directly.

b) They can be called using the dot (.) operator.

c) They have a 'this' pointer pointing to the current object.

d) They are invoked using the class name without creating an instance of the class.

Practice Questions

1. Write a program to calculate the area of circles using static data members and static member function.
2. Write a program to keep track of the number of objects using static data members.
3. Write a program to calculate the factorial of a number using a static member function.

THANK YOU!

ANY DOUBTS?