* A const object can only call const functions. To fix the error, we must make getValue() a const function.
* Like structures, class objects can be initialized.
* Unlike new, malloc() doesn't call constructor (See [this](http://www.geeksforgeeks.org/malloc-vs-new/)) If we replace malloc() with new, the constructor is called.

**1.When do we use Initializer List in C++?**

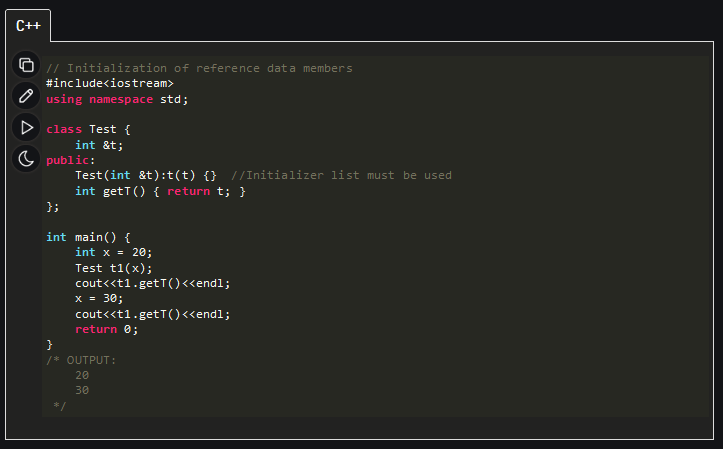
Initializer List is used in initializing the data members of a class. The list of members to be initialized is indicated with constructor as a comma-separated list followed by a colon. Following is an example that uses the initializer list to initialize x and y of Point class.

Take a step-up from those "Hello World" programs. Learn to implement data structures like Heap, Stacks, Linked List and many more! Check out our [**Data Structures in C**](https://practice.geeksforgeeks.org/courses/ds-in-c??utm_source=GeeksforGeeks&utm_medium=Text&utm_campaign=GFG_Article_Bottom_Text_DS_C) course to start learning today.

|  |
| --- |
|  |

The above code is just an example for syntax of the Initializer list. In the above code, x and y can also be easily initialed inside the constructor. But there are situations where initialization of data members inside constructor doesn’t work and Initializer List must be used. Following are such cases:  
**1) For initialization of non-static const data members:**   
const data members must be initialized using Initializer List. In the following example, “t” is a const data member of Test class and is initialized using Initializer List. Reason for initializing the const data member in initializer list is because no memory is allocated separately for const data member, it is folded in the symbol table due to which we need to initialize it in the initializer list.   
Also, it is a Parameterised constructor and we don’t need to call the assignment operator which means we are avoiding one extra operation. 

|  |
| --- |
|  |

**2) For initialization of reference members:**   
Reference members must be initialized using Initializer List. In the following example, “t” is a reference member of Test class and is initialized using Initializer List.  
****

**3) For initialization of member objects which do not have default constructor:**   
In the following example, an object “a” of class “A” is data member of class “B”, and “A” doesn’t have default constructor. Initializer List must be used to initialize “a”.

|  |
| --- |
|  |

If class A had both default and parameterized constructors, then Initializer List is not must if we want to initialize “a” using default constructor, but it is must to initialize “a” using parameterized constructor. 

**4) For initialization of base class members :** Like point 3, the parameterized constructor of the base class can only be called using Initializer List.

|  |
| --- |
|  |
| **5) When constructor’s parameter name is same as data member**  If constructor’s parameter name is same as data member name then the data member must be initialized either using [this pointer](http://msdn.microsoft.com/en-us/library/y0dddwwd.aspx) or Initializer List. In the following example, both member name and parameter name for A() is “i”. |

|  |
| --- |
|  |

**6) For Performance reasons:**   
It is better to initialize all class variables in Initializer List instead of assigning values inside body. Consider the following example:

|  |
| --- |
|  |

Here compiler follows following steps to create an object of type MyClass   
1. Type’s constructor is called first for “a”.

2. The assignment operator of “Type” is called inside body of MyClass() constructor to assign

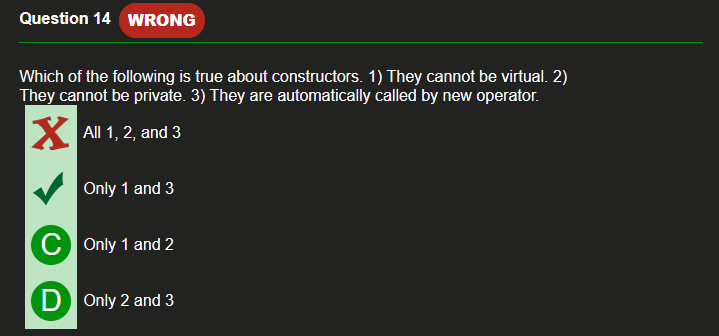
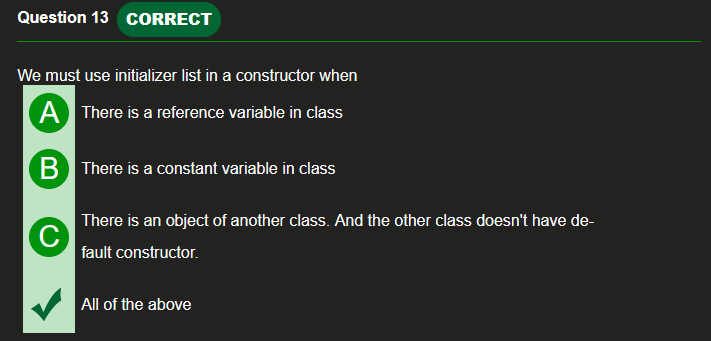
variable = a;

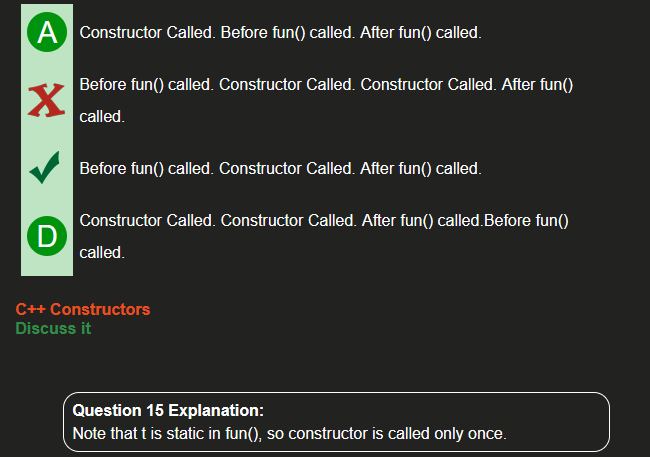
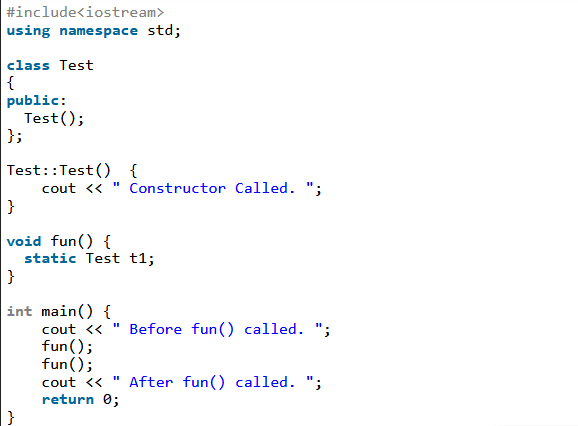
3. And then finally destructor of “Type” is called for “a” since it goes out of scope.  
Now consider the same code with MyClass() constructor with Initializer List 

|  |
| --- |
|  |

With the Initializer List, the following steps are followed by compiler:   
1. Parameterised constructor of “Type” class is called to initialize: variable(a). The arguments in the initializer list are used to copy construct “variable” directly.   
2. The destructor of “Type” is called for “a” since it goes out of scope.  
As we can see from this example if we use assignment inside constructor body there are three function calls: constructor + destructor + one addition assignment operator call. And if we use Initializer List there are only two function calls: copy constructor + destructor call. See [this](https://www.geeksforgeeks.org/output-of-c-program/) post for a running example on this point.  
This assignment penalty will be much more in “real” applications where there will be many such variables. Thanks to *ptr* for adding this point.   
Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

//THE END //





* In C++, if a class has a constructor which can be called with a single argument, then this constructor becomes conversion constructor because such a constructor allows automatic conversion to the class being constructed.

