Output of C++ Program | Set 13

Predict the output of following C++ program.

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

class A
{
    // data members of A
public:
    A ()           { cout << "\n A's constructor"; /* Initialize data members */ }
    A (const A &a) { cout << "\n A's Copy constructor";  /* copy data members */}
    A& operator= (const A &a) // Assignemt Operator
    {
        // Handle self-assignment:
        if(this == &a) return *this;

        // Copy data members
        cout << "\n A's Assignment Operator";  return *this;
    }
};

class B
{
    A a;
    // Other members of B
public:
    B(A &a) { this->a = a; cout << "\n B's constructor"; }
};

int main()
{
    A a1;
    B b(a1);
    return 0;
}
```

```
A's constructor
A's constructor
```

```
A's Assignment Operator
B's constructor
```

Output:

```
A's constructor
A's constructor
A's Assignment Operator
B's constructor
```

The first line of output is printed by the statement “A a1;” in main().
The second line is printed when B’s member ‘a’ is initialized. This is important.
The third line is printed by the statement “this->a = a;” in B’s constructor.
The fourth line is printed by cout statement in B’s constructor.

If we take a closer look at the above code, the constructor of class B is not efficient as member ‘a’ is first constructed with default constructor, and then the values from the parameter are copied using assignment operator. It may be a concern when class A is big, which generally is the case with many practical classes. See the following optimized code.

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

class A
{
    // data members of A
public:
    A()           { cout << "\n A's constructor"; /* Initialize data members */ }
    A(const A &a) { cout << "\n A's Copy constructor"; /* Copy data members */ }
    A& operator= (const A &a) // Assignemt Operator
    {
        // Handle self-assignment:
        if(this == &a) return *this;

        // Copy data members
        cout << "\n A's Assignment Operator";  return *this;
    }
};

class B
{
    A a;
    // Other members of B
public:
    B(A &a):a(a) {  cout << "\n B's constructor"; }
};

int main()
{
    A a;
    B b(a);
    return 0;
}
```

Output:

```
A's constructor
A's Copy constructor
B's constructor
```

The constructor of class B now uses initializer list to initialize its member 'a'. When Initializer list is used, the member 'a' of class B is initialized directly from the parameter. So a call to A's constructor is reduced.
In general, it is a good idea to use Initializer List to initialize all members of a class, because it saves one extra assignment of members. See point 6 of this post for more details.

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

Recommended Posts:

Output of C++ Program | Set 20
Output of C++ Program | Set 19
Output of C Program | Set 29
Output of C Program | Set 19
Output of C Program | Set 18
Output of C++ Program | Set 3
Output of C++ Program | Set 2
Output of C Program | Set 17
Output of C++ Program | Set 1
Output of C Program | Set 20
Output of C++ Program | Set 10
Output of C++ Program | Set 7
Output of C Program | Set 21
Output of C++ Program | Set 8
Output of C++ Program | Set 11

Improved By : Shun Xian Cai

Article Tags : Program Output CPP-Output

Be the First to upvote.

3.3

To-do Done

Based on **19** vote(s)

Feedback/ Suggest Improvement | 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