Destructors in C++

Destructors in C++ are members functions in a class that delete an object. They are called when the class object goes out of scope such as when the function ends, the program ends, a delete variable is called etc.

Destructors are different from normal member functions as they don’t take any argument and don’t return anything. Also, destructors have the same name as their class and their name is preceded by a tilde(~).

A program that demonstrates destructors in C++ is given as follows.

Example

Live Demo

```
#include<iostream>
using namespace std;
class Demo {
   private:
   int num1, num2;
   public:
   Demo(int n1, int n2) {
      cout<<"Inside Constructor"<<endl;
      num1 = n1;
      num2 = n2;
   }
   void display() {
      cout<<"num1 = "<< num1 <<endl;
      cout<<"num2 = "<< num2 <<endl;
   }
   ~Demo() {
      cout<<"Inside Destructor";
   }
};
int main() {
   Demo obj1(10, 20);
   obj1.display();
   return 0;
}
```

Output

```
Inside Constructor
num1 = 10
num2 = 20
Inside Destructor
```

In the above program, the class Demo contains a parameterized constructor that initializes num1 and num2 with the values provided by n1 and n2. It also contains a function display() that prints the value of num1 and num2. There is also a destructor in Demo that is called when the scope of the class object is ended. The code snippet for this is given as follows.

```
class Demo {
   private:
   int num1, num2;
   public:
   Demo(int n1, int n2) {
      cout<<"Inside Constructor"<<endl;
      num1 = n1;
      num2 = n2;
   }
   void display() {
      cout<<"num1 = "<< num1 <<endl;
      cout<<"num2 = "<< num2 <<endl;
   }
   ~Demo() {
      cout<<"Inside Destructor";
   }
};
```

The function main() contains the object definition for an object of class type Demo. Then the function display() is called. This is shown below.

```
Demo obj1(10, 20);
obj1.display();
```