

Destructor

Ali Haider

syedalihaider.ciit@gmail.com

Department of Computer Science IUB

Overview

- What is Destructor
- Syntax for Destructor
- Examples

Destructor

- the constructor—is called automatically when an object is first created.
- You might guess that another function is called automatically when an object is destroyed.
- This is indeed the case. Such a function is called a destructor.
- **When is destructor called?**
A destructor function is called automatically when the object goes out of scope:
 - (1) the function ends
 - (2) the program ends
 - (3) a block containing local variables ends
 - (4) a delete operator is called

Cont...

- **Can there be more than one destructor in a class?**
No, there can only one destructor in a class with class name preceded by ~, no parameters and no return type.
- **When do we need to write a user-defined destructor?**
If we do not write our own destructor in class, compiler creates a default destructor for us.
- The default destructor works fine unless we have dynamically allocated memory or pointer in class.
- When a class contains a pointer to memory allocated in class, we should write a destructor to release memory before the class instance is destroyed.
- This must be done to avoid memory leak.

Syntax of Destructor

- A destructor has the same name as the constructor (which is the same as the class name) but is preceded by a tilde:
- `Foo() : data(0) //constructor (same name as class)`
- `{ }`
- `~Foo() //destructor (same name with tilde)`
- `{ }`

Examples

```
C:\Users\ShahTab\Documents\OOP\Destructor.exe
Constructor called
Constructor called
Value of Num is99
Value of Num is300
Destructor called
Destructor called
-----
Process exited after 0.293 seconds with return value 0
Press any key to continue . . .
```

Output

```
#include<iostream>
using namespace std;
class Sample
{
    private:
        int num;
    public:
        // constructor
        Sample(int n):num(n){
            cout<<"Constructor called"<<endl;
        }
        // destructor
        ~Sample(){
            cout<<"Destructor called"<<endl;
        }
        void show(){
            cout<<"Value of Num is"<<num<<endl;
        }
};
int main(){
    Sample z1(99);    // Constructor Called
    Sample z2(300);   // Constructor Called
        z1.show();
        z2.show();
        // Destructor Called for z2
        // Destructor called for z1
}
```