Class- Constructor and Destructor

The Class Constructor

A class constructor is a special member function of a class that is executed whenever we create new objects of that class.

A constructor will have exact same name as the class and it does not have any return type at all, not even void. Constructors can be very useful for setting initial values for certain member variables.

Parameterized Constructor

A default constructor does not have any parameter, but if you need, a constructor can have parameters.

The Class Destructor

A destructor is a special member function of a class that is executed whenever an object of it's class goes out of scope or whenever the delete expression is applied to a pointer to the object of that class.

A destructor will have exact same name as the class prefixed with a tilde (~) and it can neither return a value nor can it take any parameters. Destructor can be very useful for releasing resources before coming out of the program like closing files, releasing memories etc.

Copy Constructor

A copy constructor is a member function which initializes an object using another object of the same class. A copy constructor has the following general function prototype:

```
ClassName (const ClassName &old_obj);
```

In C++, a Copy Constructor may be called in following cases:

- When an object of the class is returned by value.
- When an object of the class is passed (to a function) by value as an argument.
- When an object is constructed based on another object of the same class.
- When the compiler generates a temporary object.

Example

```
#include<iostream>
using namespace std;
class Point
{
private:
    int x,y;
public:
    Point(void){
        cout << "Constructiong Point" << endl;
        x=y=0;
    }
    Point(int _x, int _y){
        cout << "Constructiong Point" << endl;
        x=_x ;
        y=_y;
```

```
    }
    Point(const Point &p2){
        x = p2.x; y = p2.y;
    }
    ~Point(void) {
        cout << "Destructing Point" << endl;
    }
    void setx(int _x) { x=_x; }
    void sety(int _y) { y=_y; }
    int getx(void) const { return x; }
    int gety(void) const { return y; }
};
int main()
{
Point p1;
cout << "("  << p1.getx() <<  ","   << p1.gety() << ")" << endl;
Point p2(4,6);
cout << "("  << p2.getx() <<  ","   << p2.gety() << ")" << endl;
return 0;
}
```