DESTRUCTORS

What is destructor?

Destructor is a member function which destructs or deletes an object.

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

How destructors are different from a normal member function?

Destructors have same name as the class preceded by a tilde ~

Destructors don't take any argument and don't return anything

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.

```
class String {
private:
   char *s;
   int size;
public:
   String(char *); // constructor
   ~String();    // destructor
};

String::String(char *c) {
   size = strlen(c);
   s = new char[size+1];
   strcpy(s,c);
}

String::~String() {
   delete []s;
}
```