Lecture 5.2

Object Oriented Concepts

Constructor and Destructor Inheritance

- Constructors, and destructors are NOT inherited.
- Each subtype has its own constructor and destructor.
- Each object of a subtype consists of multiple parts, a base class part and a subclass part.
- The base class constructor forms the base class part.
- The subclass constructor forms the subclass part.

Constructor and Destructor inheritance

- Destructors clean up their respective parts.

- Let’s see an example with print statements in constructors and destructors to better see how objects are created and destroyed.

Protected Members of a Class

- Private members of a class
  - Are private
  - Cannot be accessed outside of the class
  - Even a derived class cannot directly access the private members of a base class
- Protected members –
  - Base class to give access to a member to its derived class

Friend functions of classes

- Function defined outside the scope of a class
- Is a non-member function of the class, but has access to class's private data members.
- The reserved word **friend** precedes the function prototype in class definition
- Normally used in operator overloading

friend functions of classes

- Declaration can be placed within private, protected or public part of the class

```
class Student {
   friend void IcanAccessPrivateMembers(…);
   ..
   ..
   ..
};
```

Definition of a friend function

- Name and scope resolution operator of the class do not precede the name of the friend function heading
- Also the word friend does not appear in the heading of the function.
  void IcanAccessPrivateMembers(…);
- However it is implemented in the implementation file