Public, Protected & Private Inheritance
Back to Inheritance

• Before we saw public inheritance of the form
  class Student : public person {} 
• We will now see protected and private inheritance as well.
• They differ in how the inherited members of Student are accessed by classes that are derived from Student.
• Note: Public inheritance is the norm. Private and protected inheritance are very unusual.
Example: person.h

class Person {
private:
    string name;
    string address;
    Department dept;
protected:
    void set_name(const char *name);
    void set_address(const char *adr);
    void set_department(Department dept);
public:
    Person(string n, string a, Department d) :
        name(n), address(a), dept(d) { }
    string get_name() const;
    string get_address() const;
    Department get_department() const;
};
Example: student.h

class Student : Person
{ private:
    Course *enrolled;
    int num_courses;
public:
    Student(string n, string a, Department d) :
        Person(n, a, d), enrolled(NULL), num_courses(0) { }
    bool enroll_course(const string &);
    bool drop_course(const Course &);
};

• If GraduateStudent is derived from Student what members of Student can GraduateStudent access?
Example: Public Inheritance

class Student: public Person { … }
Example: Protected Inheritance

class Student: **protected** Person { … }

<table>
<thead>
<tr>
<th>public</th>
<th>protected</th>
<th>private</th>
<th>not accessible</th>
</tr>
</thead>
<tbody>
<tr>
<td>enroll_course()</td>
<td>set_name()</td>
<td>enrolled</td>
<td>name</td>
</tr>
<tr>
<td>drop_course()</td>
<td>set_address()</td>
<td>num_courses</td>
<td>address</td>
</tr>
<tr>
<td>set_department()</td>
<td></td>
<td></td>
<td>dept</td>
</tr>
<tr>
<td>get_name()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_address()</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_department()</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example: Private Inheritance

```java
class Student: private Person { … }
```

<table>
<thead>
<tr>
<th>public</th>
<th>protected</th>
<th>private</th>
<th>not accessible</th>
</tr>
</thead>
<tbody>
<tr>
<td>enroll_course()</td>
<td>enrolled</td>
<td>name</td>
<td>name</td>
</tr>
<tr>
<td>drop_course()</td>
<td>num_courses</td>
<td>address</td>
<td>address</td>
</tr>
<tr>
<td>set_name()</td>
<td>set_address()</td>
<td>dept</td>
<td>dept</td>
</tr>
<tr>
<td>set_department()</td>
<td></td>
<td>get_name()</td>
<td>get_name()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>get_address()</td>
<td>get_address()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>get_department()</td>
<td>get_department()</td>
</tr>
</tbody>
</table>
Summary

• **Public** inheritance preserves the original accessibility of the base class’ public and protected members:
  
  public => public
  protected => protected
  private => not accessible

• **Protected** inheritance causes the accessibility of inherited public and protected members to be protected.
  
  public => protected
  protected => protected
  private => not accessible
Summary…

- **Private** inheritance renders all inherited members private.
  - public => private
  - protected => private
  - private => not accessible
- Remember that the base class’ private members are **never** accessible to derived classes (or to any other classes).
- The various types of inheritance control the highest accessibility of the inherited member data and functions.
- Public inheritance is the most common form of inheritance; protected and private inheritance are rarely used these days.