

객체지향프로그래밍 II



Lecture 5

제12장 클래스 상속 (Part 1)

1. Introduction
2. Base Classes and Derived Classes
3. `protected` Members





1. Introduction



상속 (Inheritance) ?

- ✓ 소프트웨어 재사용을 위한 효과적인 방법
- ✓ 기존 클래스로부터 새로운 클래스 생성
 - 기존 클래스의 속성과 행동 방식을 흡수
 - 새로운 속성과 행동 방식을 추가하여 보다 구체화 함
- ✓ 기본 클래스로부터 상속받아 파생 클래스가 생성된다.
 - 객체들의 좀 더 특화된 (specialized) 그룹
 - 기본 클래스(base class)로부터 행동 방식을 상속 받음
 - 자신만의 행동 방식으로 특화(customize) 가능
 - 추가적인 행동 방식도 포함

클래스 계층 (class hierarchy)

- ☞ 직접 기본 클래스 (direct base class)
 - ✓ 직접 상속 받은 바로 위 단계 기본 클래스
- ☞ 간접 기본 클래스 (indirect base class)
 - ✓ 두 단계 이상 상위의 기본 클래스
- ☞ 단일 상속 (single inheritance)
 - ✓ 하나의 기본 클래스로부터 상속
- ☞ 다중 상속 (multiple inheritance)
 - ✓ 두 개 이상의 기본 클래스로부터 상속
 - 기본 클래스간 관련이 없을 수도 있다.

상속의 세 종류

🚗 public

✓ 파생 클래스의 객체는 기본 클래스의 객체이기도 하다.

- 기본 클래스의 객체는 파생 클래스의 객체가 될 수 없다.
- 예: 모든 자동차는 탈 것이다, 그러나 모든 탈 것은 차가 아니다.

✓ 기본 클래스의 non-private 멤버에 접근 가능하다.

- 기본 클래스의 **private** 멤버에 접근하기 위해서는
 - 파생 클래스는 반드시 상속받은 non-private 멤버 함수를 사용해야 한다.

🚗 **private, protected** : 잘 사용하지 않으므로 본 강의에서는 skip 함.

상속 관계의 의미

☞ 추상화 (abstraction)

- ✓ 시스템의 모든 객체들의 공통성에 주목

☞ "*is-a*" 관계 vs. "*has-a*" 관계

- ✓ "*is-a*" 관계 - 상속

- 파생 클래스의 객체는 기본 클래스의 객체처럼 다루어진다.
- 예: 차는 탈것이다. (Car *is a* vehicle)

- ✓ "*has-a*" 관계 - 복합 (composition) 관계

- 객체가 다른 클래스의 객체를 멤버로 포함하는 것
- 예: 차에는 운전대가 있다. (Car *has a* steering wheel)



2. Base (기본) Classes and Derived (파생) Classes



기본 클래스는 파생 클래스보다 더 넓은 범위의 객체를 표현

✓ 예

- 기본 클래스: **Vehicle**
 - 차(car), 트럭, 보트, 자전거 등을 포함
- 파생 클래스: **Car**
 - 작고 보다 특화된 탈것(vehicle)의 종류

Base class	Derived classes
Student	GraduateStudent, UndergraduateStudent
Shape	Circle, Triangle, Rectangle, Sphere, Cube
Loan	CarLoan, HomeImprovementLoan, MortgageLoan
Employee	Faculty, Staff
Account	CheckingAccount, SavingsAccount

상속 계층 구조 (Inheritance hierarchy)

- ✓ 상속 관계는 트리(tree)형태의 계층 구조를 형성한다.
- ✓ 각 클래스는 다음과 같다.

기본 클래스 (base class)

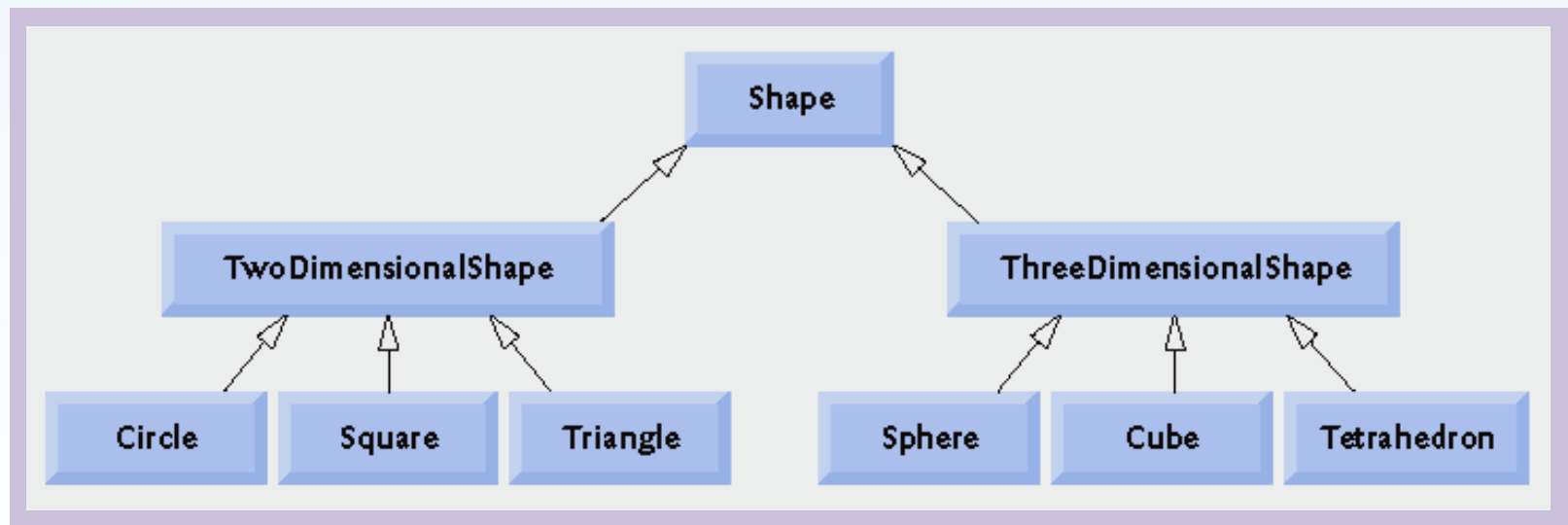
: 파생 클래스에 속성/행동 방식을 공급한다.

➤ 또는(or) ➤

파생 클래스 (derived class)

: 기본 클래스로부터 속성/행동 방식을 상속받는다.

Inheritance hierarchy for Shape



public 상속

✓ 예시

```
Class TwoDimensionalShape : public Shape
```

: TwoDimensionalShape 클래스는 Shape 클래스로부터 상속받음

✓ 기본 클래스 **private** 멤버

- 직접적으로 접근할 수 없지만 상속된다.
 - 상속된 **public** 멤버 함수를 통해 접근할 수 있다.

✓ 기본 클래스의 **public** 과 **protected** 멤버

- 파생 클래스에서 직접 접근할 수 있다.



3. `protected` Members



접근 지정자 protected

protected 접근

- ✓ public 과 private 의 중간 보호 단계
- ✓ protected 멤버는 다음을 통해 접근 가능하다.

- 기본 클래스의 멤버와 파생 클래스의 멤버

파생 클래스 멤버

- ✓ 기본 멤버의 public 과 protected 멤버에 직접 접근 가능

- 간단히 멤버 이름을 사용

- ✓ 파생 클래스에서 재정의된 기본 클래스 멤버는 기본 클래스 이름과 이항 스코프 식별 연산자 (::) 를 이용하여 접근할 수 있다.

- 예) shape::area

객체지향프로그래밍 II



Lecture 5

제12장 클래스 상속 (Part 2)

1. **CommissionEmployee** Class
2. **BasePlusCommissionEmployee** Class
WITHOUT Using Inheritance
3. Creating a Inheritance Hierarchy



기본 클래스와 파생 클래스 예제 개요

☞ CommisionEmployee - BasePlusCommissionEmployee 상속 계층

✓ CommissionEmployee

- First name, last name, SSN, commission rate, gross sale amount

✓ BasePlusCommissionEmployee

- First name, last name, SSN, commission rate, gross sale amount
- + 기본 봉급

(주) Commission Employee: 판매 금액의 일부를 수당으로 가져가는 영업 사원
(기본 월급이 있을 수도, 없을 수도 있음)



1. `CommissionEmployee` Class



CommissionEmployee 클래스 구성

☞ CommissionEmployee 헤더파일 (다음 예제)

✓ Public services

- 생성자
- `get` 과 `set` 함수
- 멤버 함수 `earnings`, `print`

☞ CommissionEmployee 소스 코드 파일 (다음 예제)

CommissionEmployee 클래스 예제 (CommissionEmployee.h)


```
1 // Fig. 12.4: CommissionEmployee.h
2 // CommissionEmployee class definition represents a commission employee.
3 #ifndef COMMISSION_H
4 #define COMMISSION_H
5
6 #include <string> // C++ standard string class
7 using std::string;
8
9 class CommissionEmployee
10 {
11 public:
12     CommissionEmployee( const string &, const string &, const string &,
13                        double = 0.0, double = 0.0 );
14
15     void setFirstName( const string & ); // set first name
16     string getFirstName() const; // return first name
17
18     void setLastName( const string & ); // set last name
19     string getLastName() const; // return last name
20
21     void setSocialSecurityNumber( const string & ); // set SSN
22     string getSocialSecurityNumber() const; // return SSN
23
24     void setGrossSales( double ); // set gross sales amount
25     double getGrossSales() const; // return gross sales amount
26
27     void setCommissionRate( double ); // set commission rate (percentage)
28     double getCommissionRate() const; // return commission rate
```

Class **CommissionEmployee** constructor

CommissionEmployee 클래스 예제 (CommissionEmployee.h)

```
29
30  double earnings() const; // calculate earnings
31  void print() const; // print CommissionEmployee object
32 private:
33  string firstName;
34  string lastName;
35  string socialSecurityNumber;
36  double grossSales; // gross weekly sales
37  double commissionRate; // commission percentage
38 }; // end class CommissionEmployee
39
40 #endif
```

Declare **private**
data members



CommissionEmployee 클래스 예제 (CommissionEmployee.cpp)

```
1 // Fig. 12.5: CommissionEmployee.cpp
2 // Class CommissionEmployee member-function definitions.
3 #include <iostream>
4 using std::cout;
5
6 #include "CommissionEmployee.h" // CommissionEmployee class definition
7
8 // constructor
9 CommissionEmployee::CommissionEmployee(
10     const string &first, const string &last, const string &ssn,
11     double sales, double rate )
12 {
13     firstName = first; // should validate
14     lastName = last;   // should validate
15     socialSecurityNumber = ssn; // should validate
16     setGrossSales( sales ); // validate and store gross sales
17     setCommissionRate( rate ); // validate and store commission rate
18 } // end CommissionEmployee constructor
19
20 // set first name
21 void CommissionEmployee::setFirstName( const string &first )
22 {
23     firstName = first; // should validate
24 } // end function setFirstName
25
26 // return first name
27 string CommissionEmployee::getFirstName() const
28 {
29     return firstName;
30 } // end function getFirstName
```

Initialize data members



CommissionEmployee 클래스 예제 (CommissionEmployee.cpp)

```
31
32 // set last name
33 void CommissionEmployee::setLastName( const string &last )
34 {
35     lastName = last; // should validate
36 } // end function setLastName
37
38 // return last name
39 string CommissionEmployee::getLastName() const
40 {
41     return lastName;
42 } // end function getLastName
43
44 // set social security number
45 void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
46 {
47     socialSecurityNumber = ssn; // should validate
48 } // end function setSocialSecurityNumber
49
50 // return social security number
51 string CommissionEmployee::getSocialSecurityNumber() const
52 {
53     return socialSecurityNumber;
54 } // end function getSocialSecurityNumber
55
56 // set gross sales amount
57 void CommissionEmployee::setGrossSales( double sales )
58 {
59     grossSales = ( sales < 0.0 ) ? 0.0 : sales;
60 } // end function setGrossSales
```

Function **setGrossSales**
validates gross sales amount



CommissionEmployee 클래스 예제 (CommissionEmployee.cpp)

```
61
62 // return gross sales amount
63 double CommissionEmployee::getGrossSales() const
64 {
65     return grossSales;
66 } // end function getGrossSales
67
68 // set commission rate
69 void CommissionEmployee::setCommissionRate( double rate )
70 {
71     commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
72 } // end function setCommissionRate
73
74 // return commission rate
75 double CommissionEmployee::getCommissionRate() const
76 {
77     return commissionRate;
78 } // end function getCommissionRate
```

Function **setCommissionRate**
validates commission rate



CommissionEmployee 클래스 예제 (CommissionEmployee.cpp)

```
79
80 // calculate earnings
81 double CommissionEmployee::earnings() const
82 {
83     return commissionRate * grossSales;
84 } // end function earnings
85
86 // print CommissionEmployee object
87 void CommissionEmployee::print() const
88 {
89     cout << "commission employee: " << firstName << ' ' << 1
90         << "\nsocial security number: " << socialSecurityNumber
91         << "\ngross sales: " << grossSales
92         << "\ncommission rate: " << commissionRate;
93 } // end function print
```

Function **earnings**
calculates earnings

Function **print** displays
CommissionEmployee object

CommissionEmployee 클래스 예제 (driver)

```
1 // Fig. 12.6: fig12_06.cpp
2 // Testing class CommissionEmployee.
3 #include <iostream>
4 using std::cout;
5 using std::endl;
6 using std::fixed;
7
8 #include <iomanip>
9 using std::setprecision;
10
11 #include "CommissionEmployee.h" // CommissionEmployee class definition
12
13 int main()
14 {
15     // instantiate a CommissionEmployee object
16     CommissionEmployee employee(
17         "Sue", "Jones", "222-22-2222", 10000, .06 );
18
19     // set floating-point output formatting
20     cout << fixed << setprecision( 2 );
21
22     // get commission employee data
23     cout << "Employee information obtained by get functions: \n"
24         << "\nFirst name is " << employee.getFirstName()
25         << "\nLast name is " << employee.getLastName()
26         << "\nSocial security number is "
27         << employee.getSocialSecurityNumber()
28         << "\nGross sales is " << employee.getGrossSales()
29         << "\nCommission rate is " << employee.getCommissionRate() << endl;
```

Instantiate **CommissionEmployee** object

Use **CommissionEmployee**'s
get functions to retrieve the
object's instance variable values

CommissionEmployee 클래스 예제 (driver 및 실행결과)

```
30
31 employee.setGrossSales( 8000 ); // set gross sales
32 employee.setCommissionRate( .1 ); // set commission rate
33
34 cout << "\nUpdated employee information output
35     << endl;
36 employee.print(); // display the new employee information
37
38 // display the employee's earnings
39 cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
40
41 return 0;
42 } // end main
```

Use **CommissionEmployee**'s *set* functions to change the object's instance variable values

Call object's **print** function to display employee information

Call object's **earnings** function to calculate earnings

Employee information obtained by get functions:

First name is Sue
Last name is Jones
Social security number is 222-22-2222
Gross sales is 10000.00
Commission rate is 0.06

Updated employee information output by print function:

commission employee: Sue Jones
social security number: 222-22-2222
gross sales: 8000.00
commission rate: 0.10

Employee's earnings: \$800.00



2. BasePlusCommissionEmployee Class WITHOUT Using Inheritance



BasePlusCommissionEmployee 클래스 Ver. 1

- ✓ 상속 없이 속성을 추가하는 것이 얼마나 비효율적인지 파악하기 위함
- ✓ 코드의 대부분이 CommissionEmployee 와 유사

- `private` 데이터 멤버
- `public` 멤버 함수
- 생성자

- ✓ 추가된 멤버

- `private` 데이터 멤버 `baseSalary`
- `setBaseSalary` 와 `getBaseSalary` 멤버 함수

- ✓ 상속을 이용한 Ver. 2와 비교할 것

상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
1 // BasePlusEmployee.h
2 // BasePlusEmployee class definition represents an employee
3 // that receives a base salary in addition to commission.
4 #ifndef BASEPLUS_H
5 #define BASEPLUS_H
6
7 #include <string> // C++ standard string class
8 using std::string;
9
10 class BasePlusCommissionEmployee
11 {
12 public:
13     BasePlusCommissionEmployee( const string &, const string &,
14                                const string &, double = 0.0, double = 0.0, double = 0.0 );
15
16     void setFirstName( const string & ); // set first name
17     string getFirstName() const; // return first name
18
19     void setLastName( const string & ); // set last name
20     string getLastName() const; // return last name
21
22     void setSocialSecurityNumber( const string & ); // set SSN
23     string getSocialSecurityNumber() const; // return SSN
24
25     void setGrossSales( double ); // set gross sales amount
26     double getGrossSales() const; // return gross sales amount
27
28     void setCommissionRate( double ); // set commission rate
29     double getCommissionRate() const; // return commission rate
```

Constructor takes one more argument, which specifies the base salary

상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
30
31 void setBaseSalary( double ); // set base salary
32 double getBaseSalary() const; // return base salary
33
34 double earnings() const; // calculate earnings
35 void print() const; // print BasePlusCommissionEmployee object
36 private:
37     string firstName;
38     string lastName;
39     string socialSecurityNumber;
40     double grossSales; // gross weekly sales
41     double commissionRate; // commission percentage
42     double baseSalary; // base salary
43 }; // end class BasePlusCommissionEmployee
44
45 #endif
```

Define *get* and *set* functions for data member **baseSalary**

Add data member **baseSalary**

상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
1 // Fig. 12.8: BasePlusCommissionEmployee.cpp
2 // Class BasePlusCommissionEmployee member-function definitions.
3 #include <iostream>
4 using std::cout;
5
6 // BasePlusCommissionEmployee class definition
7 #include "BasePlusCommissionEmployee.h"
8
9 // constructor
10 BasePlusCommissionEmployee::BasePlusCommissionEmployee(
11     const string &first, const string &last, const string &ssn,
12     double sales, double rate, double salary )
13 {
14     firstName = first; // should validate
15     lastName = last; // should validate
16     socialSecurityNumber = ssn; // should validate
17     setGrossSales( sales ); // validate and store gross sales
18     setCommissionRate( rate ); // validate and store commission rate
19     setBaseSalary( salary ); // validate and store base salary
20 } // end BasePlusCommissionEmployee constructor
21
22 // set first name
23 void BasePlusCommissionEmployee::setFirstName( const string &first )
24 {
25     firstName = first; // should validate
26 } // end function setFirstName
```

Constructor takes one more argument,
which specifies the base salary

Use function **setBaseSalary** to validate data

상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
27
28 // return first name
29 string BasePlusCommissionEmployee::getFirstName() const
30 {
31     return firstName;
32 } // end function getFirstName
33
34 // set last name
35 void BasePlusCommissionEmployee::setLastName( const string &last )
36 {
37     lastName = last; // should validate
38 } // end function setLastName
39
40 // return last name
41 string BasePlusCommissionEmployee::getLastName() const
42 {
43     return lastName;
44 } // end function getLastName
45
46 // set social security number
47 void BasePlusCommissionEmployee::setSocialSecurityNumber(
48     const string &ssn )
49 {
50     socialSecurityNumber = ssn; // should validate
51 } // end function setSocialSecurityNumber
52
```

상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
53 // return social security number
54 string BasePlusCommissionEmployee::getSocialSecurityNumber() const
55 {
56     return socialSecurityNumber;
57 } // end function getSocialSecurityNumber
58
59 // set gross sales amount
60 void BasePlusCommissionEmployee::setGrossSales( double sales )
61 {
62     grossSales = ( sales < 0.0 ) ? 0.0 : sales;
63 } // end function setGrossSales
64
65 // return gross sales amount
66 double BasePlusCommissionEmployee::getGrossSales() const
67 {
68     return grossSales;
69 } // end function getGrossSales
70
71 // set commission rate
72 void BasePlusCommissionEmployee::setCommissionRate( double rate )
73 {
74     commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
75 } // end function setCommissionRate
76
77 // return commission rate
78 double BasePlusCommissionEmployee::getCommissionRate() const
79 {
80     return commissionRate;
81 } // end function getCommissionRate
82
```


상속 없이 구현한 BasePlusCommissionEmployee 클래스

```
83 // set base salary
84 void BasePlusCommissionEmployee::setBaseSalary( double salary )
85 {
86     baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
87 } // end function setBaseSalary
88
89 // return base salary
90 double BasePlusCommissionEmployee::getBaseSalary() const
91 {
92     return baseSalary;
93 } // end function getBaseSalary
94
95 // calculate earnings
96 double BasePlusCommissionEmployee::earnings() const
97 {
98     return baseSalary + ( commissionRate * grossSales );
99 } // end function earnings
100
101 // print BasePlusCommissionEmployee object
102 void BasePlusCommissionEmployee::print() const
103 {
104     cout << "base-salaried commission employee: " << firstName << ' '
105         << lastName << "\nsocial security number: " << socialSecurityNumber
106         << "\ngross sales: " << grossSales
107         << "\ncommission rate: " << commissionRate
108         << "\nbase salary: " << baseSalary;
109 } // end function print
```

Function **setBaseSalary** validates data and sets instance variable **baseSalary**

Function **getBaseSalary** returns the value of instance variable **baseSalary**

Update function **earnings** to calculate the earnings of a base-salaried commission employee

Update function **print** to display base salary



3. Creating a Inheritance Hierarchy



BasePlusCommissionEmployee 클래스 Ver. 2

✓ `CommissionEmployee` 클래스로부터 상속받아 파생

- */s-a* `CommissionEmployee`
- 모든 `public` 멤버 상속

✓ 생성자는 상속되지 않는다.

- 기본 클래스의 데이터 멤버 초기화를 위해 기본 클래스 초기화 구문을 사용

✓ 데이터 멤버 `baseSalary` 를 갖는다.

상속을 이용하여 구현한 BasePlusCommissionEmployee 클래스

```
1 // Fig. 12.10: BasePlusCommissionEmployee.h
2 // BasePlusCommissionEmployee class derived from class
3 // CommissionEmployee.
4 #ifndef BASEPLUS_H
5 #define BASEPLUS_H
6
7 #include <string> // C++ standard string class
8 using std::string;
9
10 #include "CommissionEmployee.h" // CommissionEmployee class declaration
11
12 class BasePlusCommissionEmployee : public CommissionEmployee
13 {
14 public:
15     BasePlusCommissionEmployee( const string &, const string &,
16         const string &, double = 0.0, double = 0.0, double = 0.0 );
17
18     void setBaseSalary( double ); // set base salary
19     double getBaseSalary() const; // return base salary
20
21     double earnings() const; // calculate earnings
22     void print() const; // print BasePlusCommissionEmployee object
23 private:
24     double baseSalary; // base salary
25 }; // end class BasePlusCommissionEmployee
26
27 #endif
```

Include the base-class header file
in the derived-class header file

Class BasePlusCommissionEmployee
derives **publicly** from class
CommissionEmployee

상속을 이용하여 구현한 BasePlusCommissionEmployee 클래스

```
1 // Fig. 12.11: BasePlusCommissionEmployee.cpp
2 // Class BasePlusCommissionEmployee member-function definitions.
3 #include <iostream>
4 using std::cout;
5
6 // BasePlusCommissionEmployee class definition
7 #include "BasePlusCommissionEmployee.h"
8
9 // constructor
10 BasePlusCommissionEmployee::BasePlusCommissionEmployee(
11     const string &first, const string &last, const string &ssn,
12     double sales, double rate, double salary )
13     // explicitly call base-class constructor
14     : CommissionEmployee( first, last, ssn, sales, rate )
15 {
16     setBaseSalary( salary ); // validate and store base salary
17 } // end BasePlusCommissionEmployee constructor
18
19 // set base salary
20 void BasePlusCommissionEmployee::setBaseSalary( double salary )
21 {
22     baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
23 } // end function setBaseSalary
24
25 // return base salary
26 double BasePlusCommissionEmployee::getBaseSalary() const
27 {
28     return baseSalary;
29 } // end function getBaseSalary
```

Initialize base class data member by calling the base-class constructor using base-class initializer syntax

상속을 이용하여 구현한 BasePlusCommissionEmployee 클래스

```
30
31 // calculate earnings
32 double BasePlusCommissionEmployee::earnings() const
33 {
34     // derived class cannot access the base class's private data
35     return baseSalary + ( commissionRate * grossSales );
36 } // end function earnings
37
38 // print BasePlusCommissionEmployee object
39 void BasePlusCommissionEmployee::print() const
40 {
41     // derived class cannot access the base class's private data
42     cout << "base-salaried commission employee: " << firstName << "
43         << lastName << "\nsocial security number: " << socialSecurityNumber
44         << "\ngross sales: " << grossSales
45         << "\ncommission rate: " << commissionRate
46         << "\nbase salary: " << baseSalary;
47 } // end function print
```

Compiler generates errors because base class's data member **commissionRate** and **grossSales** are **private**

Compiler generates errors because the base class's data members **firstName**, **lastName**, **socialSecurityNumber**, **grossSales** and **commissionRate** are **private**

상속은 잘 되었으나, Base class의 Private 멤버에 접근하는 방식에서 컴파일 오류 발생 → 해결책은 다음 시간에...

Creating a Inheritance Hierarchy (cont)

☞ 기본 클래스 헤더 파일을 **include**한다.

✓ **기본 클래스**의 헤더 파일은 다음 세가지 이유로 반드시 파생 클래스 헤더 파일에 include 되어야 한다.

- 기본 클래스의 존재를 알기 위해
- 상속된 데이터 멤버의 크기를 알기 위해
- 상속된 클래스의 멤버가 올바르게 사용되는지 알기 위해