Computer Programming
Chapter 12: Inheritance – Part2

Hung-Yu Wei
Department of Electrical Engineering
National Taiwan University

Section 12.4 examples

- 12.4.1 Commission employee class (Fig 12.4~12.6)
- 12.4.2 Base+Commission employee class (Fig 12.7~12.9)
- 12.4.3 [Error] inheritance (Fig 12.10~12.11)
  - Derived class **cannot** access base class' s private data
- 12.4.4 protected data in inheritance(Fig 12.12~12.16)
  - Base class' s protected data can be accessed by
    - Member function of derived class
    - Friend of derived class
- 12.4.5 rewrite the example with Set, Get functions (Fig 12.17~12.21)
  - a better example with software engineering techniques

Section 12.4.1

Commission employee class (without inheritance)

```
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 namespace std;
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
```

Fig. 12.4 | CommissionEmployee class header file. (Part 1 of 2.)

```
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
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
```

Fig. 12.4 | `CommissionEmployee` class header file. (Part 2 of 2.)

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

Fig. 12.5 | Implementation file for `CommissionEmployee` class that represents an employee who is paid a percentage of gross sales. (Part 1 of 5.)

```
24
25   // return first name
26   string CommissionEmployee::getFirstName() const
27   {
28      return firstName;
29   } // end function getFirstName
30
31   // set last name
32   void CommissionEmployee::setLastName( const string &last )
33   {
34      lastName = last; // should validate
35   } // end function setLastName
36
37   // return last name
38   string CommissionEmployee::getLastName() const
39   {
40      return lastName;
41   } // end function getLastName
42
```

Fig. 12.5 | Implementation file for `CommissionEmployee` class that represents an employee who is paid a percentage of gross sales. (Part 2 of 5.)

```
43   // set social security number
44   void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
45   {
46      socialSecurityNumber = ssn; // should validate
47   } // end function setSocialSecurityNumber
48
49   // return social security number
50   string CommissionEmployee::getSocialSecurityNumber() const
51   {
52      return socialSecurityNumber;
53   } // end function getSocialSecurityNumber
54
55   // set gross sales amount
56   void CommissionEmployee::setGrossSales( double sales )
57   {
58      grossSales = ( sales < 0.0 ) ? 0.0 : sales;
59   } // end function setGrossSales
60
61   // return gross sales amount
62   double CommissionEmployee::getGrossSales() const
63   {
64      return grossSales;
65   } // end function getGrossSales
```

Fig. 12.5 | Implementation file for `CommissionEmployee` class that represents an employee who is paid a percentage of gross sales. (Part 3 of 5.)

```
66
67    // set commission rate
68    void CommissionEmployee::setCommissionRate( double rate )
69    {
70       commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
71    } // end function setCommissionRate
72
73    // return commission rate
74    double CommissionEmployee::getCommissionRate() const
75    {
76       return commissionRate;
77    } // end function getCommissionRate
78
79    // calculate earnings
80    double CommissionEmployee::earnings() const
81    {
82       return commissionRate * grossSales;
83    } // end function earnings
84
```

Fig. 12.5 | Implementation file for `CommissionEmployee` class that represents an employee who is paid a percentage of gross sales. (Part 4 of 5.)

```
85  // print CommissionEmployee object
86  void CommissionEmployee::print() const
87  {
88     cout << "commission employee: " << firstName << ' ' << lastName
89        << "\nsocial security number: " << socialSecurityNumber
90        << "\ngross sales: " << grossSales
91        << "\ncommission rate: " << commissionRate;
92  } // end function print
```

Fig. 12.5 | Implementation file for `CommissionEmployee` class that represents an employee who is paid a percentage of gross sales. (Part 5 of 5.)

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

Fig. 12.6 | CommissionEmployee class test program. (Part 1 of 3.)

```
25
26       employee.setGrossSales( 8000 ); // set gross sales
27       employee.setCommissionRate( .1 ); // set commission rate
28
29       cout << "\nUpdated employee information output by print function: \n"
30          << endl;
31       employee.print(); // display the new employee information
32
33       // display the employee's earnings
34       cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
35    } // end main
```

Fig. 12.6 | CommissionEmployee class test program. (Part 2 of 3.)

Section 12.4.2

Base+Commission employee class (without inheritance)

```
1  // Fig. 12.7: BasePlusCommissionEmployee.h
2  // BasePlusCommissionEmployee 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 namespace std;
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
```


Fig. 12.7 | BasePlusCommissionEmployee class header file. (Part 1 of 2.)

```
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
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
```

Fig. 12.7 | `BasePlusCommissionEmployee` class header file. (Part 2 of 2.)

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

Fig. 12.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 1 of 5.)

```
20   // set first name
21   void BasePlusCommissionEmployee::setFirstName( const string &first )
22   {
23      firstName = first; // should validate
24   } // end function setFirstName
25
26   // return first name
27   string BasePlusCommissionEmployee::getFirstName() const
28   {
29      return firstName;
30   } // end function getFirstName
31
32   // set last name
33   void BasePlusCommissionEmployee::setLastName( const string &last )
34   {
35      lastName = last; // should validate
36   } // end function setLastName
37
38   // return last name
39   string BasePlusCommissionEmployee::getLastName() const
40   {
41      return lastName;
42   } // end function getLastName
```

Fig. 12.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 2 of 5.)

```
43
44   // set social security number
45   void BasePlusCommissionEmployee::setSocialSecurityNumber(
46      const string &ssn )
47   {
48      socialSecurityNumber = ssn; // should validate
49   } // end function setSocialSecurityNumber
50
51   // return social security number
52   string BasePlusCommissionEmployee::getSocialSecurityNumber() const
53   {
54      return socialSecurityNumber;
55   } // end function getSocialSecurityNumber
56
57   // set gross sales amount
58   void BasePlusCommissionEmployee::setGrossSales( double sales )
59   {
60      grossSales = ( sales < 0.0 ) ? 0.0 : sales;
61   } // end function setGrossSales
62
```

Fig. 12.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 3 of 5.)

```
63   // return gross sales amount
64   double BasePlusCommissionEmployee::getGrossSales() const
65   {
66      return grossSales;
67   } // end function getGrossSales
68
69   // set commission rate
70   void BasePlusCommissionEmployee::setCommissionRate( double rate )
71   {
72      commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
73   } // end function setCommissionRate
74
75   // return commission rate
76   double BasePlusCommissionEmployee::getCommissionRate() const
77   {
78      return commissionRate;
79   } // end function getCommissionRate
80
81   // set base salary
82   void BasePlusCommissionEmployee::setBaseSalary( double salary )
83   {
84      baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
85   } // end function setBaseSalary
```

Fig. 12.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 4 of 5.)

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

Pay by commission & base salary

Fig. 12.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 5 of 5.)

```
1  // Fig. 12.9: fig12_09.cpp
2  // Testing class BasePlusCommissionEmployee.
3  #include <iostream>
4  #include <iomanip>
5  #include "BasePlusCommissionEmployee.h"
6  using namespace std;
7
8  int main()
9  {
10    // instantiate BasePlusCommissionEmployee object
11    BasePlusCommissionEmployee
12       employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
13
14    // set floating-point output formatting
15    cout << fixed << setprecision( 2 );
16
17    // get commission employee data
18    cout << "Employee information obtained by get functions: \n"
19       << "\nFirst name is " << employee.getFirstName()
20       << "\nLast name is " << employee.getLastName()
21       << "\nSocial security number is "
22       << employee.getSocialSecurityNumber()
23       << "\nGross sales is " << employee.getGrossSales()
```

Fig. 12.9 | BasePlusCommissionEmployee class test program. (Part 1 of 3.)

```
24          << "\nCommission rate is " << employee.getCommissionRate()
25          << "\nBase salary is " << employee.getBaseSalary() << endl;
26
27       employee.setBaseSalary( 1000 ); // set base salary
28
29       cout << "\nUpdated employee information output by print function: \n"
30          << endl;
31       employee.print(); // display the new employee information
32
33       // display the employee's earnings
34       cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
35    } // end main
```

Fig. 12.9 | BasePlusCommissionEmployee class test program. (Part 2 of 3.)

Section 12.4.5

CommissionEmployee-BasePlusCommissionEmployee Inheritance Hierarchy Using private Data

Better Software Engineering

- Use SetFunction and GetFunction

- Base class uses private data member
  - Derived class cannot access base class’ s private data member

- Derived class can call the public member function of the base class

  BaseClassName :: basePublicMemberFunction()

```
1  // Fig. 12.17: CommissionEmployee.h
2  // CommissionEmployee class definition with good software engineering.
3  #ifndef COMMISSION_H
4  #define COMMISSION_H
5
6  #include <string> // C++ standard string class
7  using namespace std;
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
```

Fig. 12.17 | `CommissionEmployee` class defined using good software engineering practices. (Part 1 of 2.)

```
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
28      double getCommissionRate() const; // return commission rate
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
```

Private data member

Fig. 12.17 | `CommissionEmployee` class defined using good software engineering practices. (Part 2 of 2.)

```
1  // Fig. 12.18: CommissionEmployee.cpp
2  // Class CommissionEmployee member-function definitions.
3  #include <iostream>
4  #include "CommissionEmployee.h" // CommissionEmployee class definition
5  using namespace std;
6
7  // constructor
8  CommissionEmployee::CommissionEmployee(
9     const string &first, const string &last, const string &ssn,
10    double sales, double rate )
11    : firstName( first ), lastName( last ), socialSecurityNumber( ssn )
12 {
13    setGrossSales( sales ); // validate and store gross sales
14    setCommissionRate( rate ); // validate and store commission rate
15 } // end CommissionEmployee constructor
16
17 // set first name
18 void CommissionEmployee::setFirstName( const string &first )
19 {
20    firstName = first; // should validate
21 } // end function setFirstName
```

Fig. 12.18 | `CommissionEmployee` class implementation file: `CommissionEmployee` class uses member functions to manipulate its `private` data. (Part 1 of 5.)

```
22
23   // return first name
24   string CommissionEmployee::getFirstName() const
25   {
26      return firstName;
27   } // end function getFirstName
28
29   // set last name
30   void CommissionEmployee::setLastName( const string &last )
31   {
32      lastName = last; // should validate
33   } // end function setLastName
34
35   // return last name
36   string CommissionEmployee::getLastName() const
37   {
38      return lastName;
39   } // end function getLastName
40
```

Fig. 12.18 | `CommissionEmployee` class implementation file: `CommissionEmployee` class uses member functions to manipulate its `private` data. (Part 2 of 5.)

```
41   // set social security number
42   void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
43   {
44      socialSecurityNumber = ssn; // should validate
45   } // end function setSocialSecurityNumber
46
47   // return social security number
48   string CommissionEmployee::getSocialSecurityNumber() const
49   {
50      return socialSecurityNumber;
51   } // end function getSocialSecurityNumber
52
53   // set gross sales amount
54   void CommissionEmployee::setGrossSales( double sales )
55   {
56      grossSales = ( sales < 0.0 ) ? 0.0 : sales;
57   } // end function setGrossSales
58
```

Fig. 12.18 | `CommissionEmployee` class implementation file: `CommissionEmployee` class uses member functions to manipulate its `private` data. (Part 3 of 5.)

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

Fig. 12.18 | `CommissionEmployee` class implementation file: `CommissionEmployee` class uses member functions to manipulate its `private` data. (Part 4 of 5.)

```
77   // calculate earnings
78   double CommissionEmployee::earnings() const
79   {
80      return getCommissionRate() * getGrossSales();
81   } // end function earnings
82
83   // print CommissionEmployee object
84   void CommissionEmployee::print() const
85   {
86      cout << "commission employee: "
87         << getFirstName() << ' ' << getLastName()
88         << "\nsocial security number: " << getSocialSecurityNumber()
89         << "\ngross sales: " << getGrossSales()
90         << "\ncommission rate: " << getCommissionRate();
91   } // end function print
```

Fig. 12.18 | `CommissionEmployee` class implementation file: `CommissionEmployee` class uses member functions to manipulate its `private` data. (Part 5 of 5.)

```
1  // Fig. 12.19: 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  #include "CommissionEmployee.h" // CommissionEmployee class declaration
9  using namespace std;
10
11 class BasePlusCommissionEmployee : public CommissionEmployee
12 {
13 public:
14    BasePlusCommissionEmployee( const string &, const string &,
15       const string &, double = 0.0, double = 0.0, double = 0.0 );
16
17    void setBaseSalary( double ); // set base salary
18    double getBaseSalary() const; // return base salary
19
20    double earnings() const; // calculate earnings
21    void print() const; // print BasePlusCommissionEmployee object
22 private:
23    double baseSalary; // base salary
24 }; // end class BasePlusCommissionEmployee
25
26 #endif
```

Fig. 12.19 | `BasePlusCommissionEmployee` class header file. (Part 2 of 2.)

```
1  // Fig. 12.20: BasePlusCommissionEmployee.cpp
2  // Class BasePlusCommissionEmployee member-function definitions.
3  #include <iostream>
4  #include "BasePlusCommissionEmployee.h"
5  using namespace std;
6
7  // constructor
8  BasePlusCommissionEmployee::BasePlusCommissionEmployee(
9     const string &first, const string &last, const string &ssn,
10    double sales, double rate, double salary )
11    // explicitly call base-class constructor
12    : CommissionEmployee( first, last, ssn, sales, rate )
13 {
14    setBaseSalary( salary ); // validate and store base salary
15 } // end BasePlusCommissionEmployee constructor
16
17 // set base salary
18 void BasePlusCommissionEmployee::setBaseSalary( double salary )
19 {
20    baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
21 } // end function setBaseSalary
22
```

Fig. 12.20 | `BasePlusCommissionEmployee` class that inherits from class `CommissionEmployee` but cannot directly access the class's `private` data. (Part 1 of 2.)

```
23   // return base salary
24   double BasePlusCommissionEmployee::getBaseSalary() const
25   {
26      return baseSalary;
27   } // end function getBaseSalary
28
29   // calculate earnings
30   double BasePlusCommissionEmployee::earnings() const
31   {
32      return getBaseSalary() + CommissionEmployee::earnings();
33   } // end function earnings
34
35   // print BasePlusCommissionEmployee object
36   void BasePlusCommissionEmployee::print() const
37   {
38      cout << "base-salaried ";
39
40      // invoke CommissionEmployee's print function
41      CommissionEmployee::print();
42
43      cout << "\nbase salary: " << getBaseSalary();
44   } // end function print
```

Fig. 12.20 | `BasePlusCommissionEmployee` class that inherits from class `CommissionEmployee` but cannot directly access the class's `private` data. (Part 2 of 2.)

```
1  // Fig. 12.21: fig12_21.cpp
2  // Testing class BasePlusCommissionEmployee.
3  #include <iostream>
4  #include <iomanip>
5  #include "BasePlusCommissionEmployee.h"
6  using namespace std;
7
8  int main()
9  {
10    // instantiate BasePlusCommissionEmployee object
11    BasePlusCommissionEmployee
12       employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
13
14    // set floating-point output formatting
15    cout << fixed << setprecision( 2 );
16
17    // get commission employee data
18    cout << "Employee information obtained by get functions: \n"
19       << "\nFirst name is " << employee.getFirstName()
20       << "\nLast name is " << employee.getLastName()
21       << "\nSocial security number is "
22       << employee.getSocialSecurityNumber()
```

Fig. 12.21 | Base-class `private` data is accessible to a derived class via `public` or `protected` member function inherited by the derived class. (Part 1 of 3.)

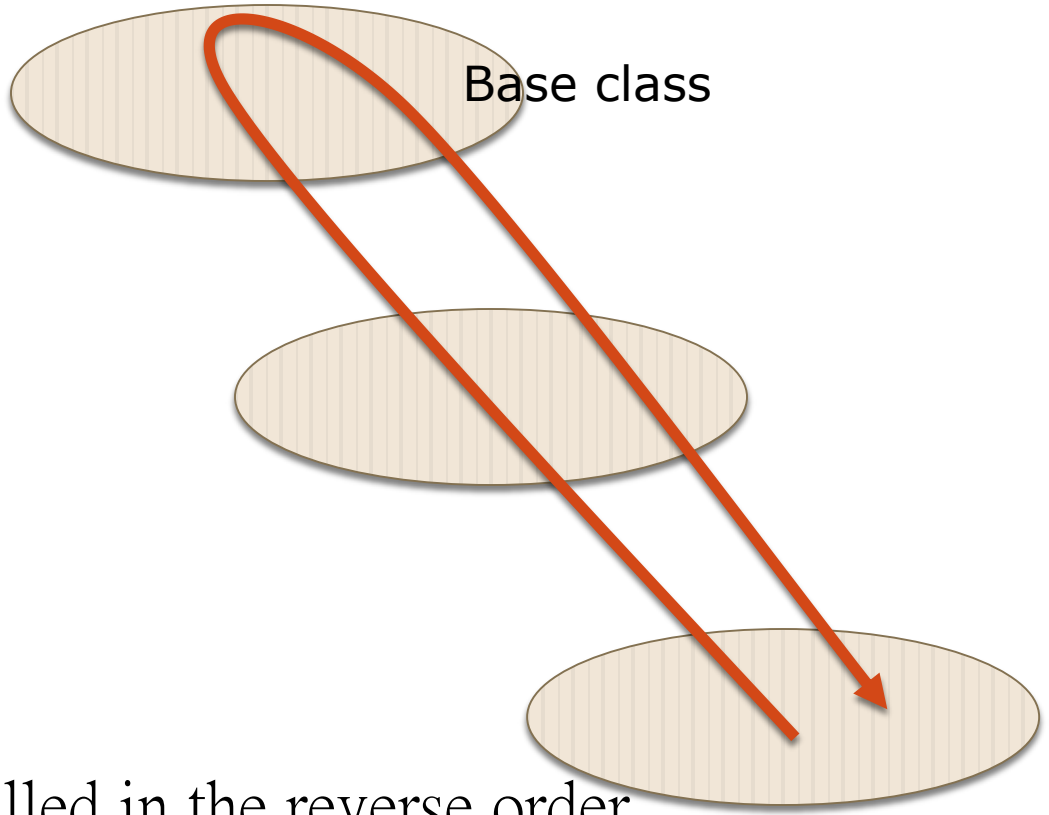
```
23          << "\nGross sales is " << employee.getGrossSales()
24          << "\nCommission rate is " << employee.getCommissionRate()
25          << "\nBase salary is " << employee.getBaseSalary() << endl;
26
27       employee.setBaseSalary( 1000 ); // set base salary
28
29       cout << "\nUpdated employee information output by print function: \n"
30          << endl;
31       employee.print(); // display the new employee information
32
33       // display the employee's earnings
34       cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
35    } // end main
```

Fig. 12.21 | Base-class `private` data is accessible to a derived class via `public` or `protected` member function inherited by the derived class. (Part 2 of 3.)

12.5 Constructor and Destructors in Derived Classes

When is constructor called?

- Constructor



- Destructor is called in the reverse order

```
1  // Fig. 12.22: 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 namespace std;
8
9  class CommissionEmployee
10 {
11 public:
12    CommissionEmployee( const string &, const string &, const string &,
13       double = 0.0, double = 0.0 );
14    ~CommissionEmployee(); // destructor
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
```

Fig. 12.22 | CommissionEmployee class header file. (Part 1 of 2.)

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

Fig. 12.22 | `CommissionEmployee` class header file. (Part 2 of 2.)

```
1  // Fig. 12.23: CommissionEmployee.cpp
2  // Class CommissionEmployee member-function definitions.
3  #include <iostream>
4  #include "CommissionEmployee.h" // CommissionEmployee class definition
5  using namespace std;
6
7  // constructor
8  CommissionEmployee::CommissionEmployee(
9     const string &first, const string &last, const string &ssn,
10    double sales, double rate )
11    : firstName( first ), lastName( last ), socialSecurityNumber( ssn )
12 {
13    setGrossSales( sales ); // validate and store gross sales
14    setCommissionRate( rate ); // validate and store commission rate
15
16    cout << "CommissionEmployee constructor: " << endl;
17    print();
18    cout << "\n\n";
19 } // end CommissionEmployee constructor
20
```

Fig. 12.23 | `CommissionEmployee`'s constructor and destructor output text. (Part 1 of 5.)

```
21  // destructor
22  CommissionEmployee::~CommissionEmployee()
23  {
24     cout << "CommissionEmployee destructor: " << endl;
25     print();
26     cout << "\n\n";
27  } // end CommissionEmployee destructor
28
29  // set first name
30  void CommissionEmployee::setFirstName( const string &first )
31  {
32     firstName = first; // should validate
33  } // end function setFirstName
34
35  // return first name
36  string CommissionEmployee::getFirstName() const
37  {
38     return firstName;
39  } // end function getFirstName
40
```

Fig. 12.23 | `CommissionEmployee`'s constructor and destructor output text. (Part 2 of 5.)

```
41   // set last name
42   void CommissionEmployee::setLastName( const string &last )
43   {
44      lastName = last; // should validate
45   } // end function setLastName
46
47   // return last name
48   string CommissionEmployee::getLastName() const
49   {
50      return lastName;
51   } // end function getLastName
52
53   // set social security number
54   void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
55   {
56      socialSecurityNumber = ssn; // should validate
57   } // end function setSocialSecurityNumber
58
59   // return social security number
60   string CommissionEmployee::getSocialSecurityNumber() const
61   {
62      return socialSecurityNumber;
63   } // end function getSocialSecurityNumber
```

Fig. 12.23 | `CommissionEmployee`'s constructor and destructor output text. (Part 3 of 5.)

```
64
65   // set gross sales amount
66   void CommissionEmployee::setGrossSales( double sales )
67   {
68      grossSales = ( sales < 0.0 ) ? 0.0 : sales;
69   } // end function setGrossSales
70
71   // return gross sales amount
72   double CommissionEmployee::getGrossSales() const
73   {
74      return grossSales;
75   } // end function getGrossSales
76
77   // set commission rate
78   void CommissionEmployee::setCommissionRate( double rate )
79   {
80      commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
81   } // end function setCommissionRate
82
```

Fig. 12.23 | `CommissionEmployee`'s constructor and destructor output text. (Part 4 of 5.)

```
83   // return commission rate
84   double CommissionEmployee::getCommissionRate() const
85   {
86      return commissionRate;
87   } // end function getCommissionRate
88
89   // calculate earnings
90   double CommissionEmployee::earnings() const
91   {
92      return getCommissionRate() * getGrossSales();
93   } // end function earnings
94
95   // print CommissionEmployee object
96   void CommissionEmployee::print() const
97   {
98      cout << "commission employee: "
99         << getFirstName() << ' ' << getLastName()
100        << "\nsocial security number: " << getSocialSecurityNumber()
101        << "\ngross sales: " << getGrossSales()
102        << "\ncommission rate: " << getCommissionRate();
103  } // end function print
```

Fig. 12.23 | `CommissionEmployee`'s constructor and destructor output text. (Part 5 of 5.)

```
1  // Fig. 12.24: 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  #include "CommissionEmployee.h" // CommissionEmployee class declaratio
9  using namespace std;
10
11 class BasePlusCommissionEmployee : public CommissionEmployee
12 {
13 public:
14    BasePlusCommissionEmployee( const string &, const string &,
15       const string &, double = 0.0, double = 0.0, double = 0.0 );
16    ~BasePlusCommissionEmployee(); // destructor
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
```

Fig. 12.24 | BasePlusCommissionEmployee class header file. (Part 1 of 2.)

```
23   private:
24      double baseSalary; // base salary
25   }; // end class BasePlusCommissionEmployee
26
27   #endif
```

Fig. 12.24 | `BasePlusCommissionEmployee` class header file. (Part 2 of 2.)

```
1  // Fig. 12.25: BasePlusCommissionEmployee.cpp
2  // Class BasePlusCommissionEmployee member-function definitions.
3  #include <iostream>
4  #include "BasePlusCommissionEmployee.h"
5  using namespace std;
6
7  // constructor
8  BasePlusCommissionEmployee::BasePlusCommissionEmployee(
9     const string &first, const string &last, const string &ssn,
10    double sales, double rate, double salary )
11    // explicitly call base-class constructor
12    : CommissionEmployee( first, last, ssn, sales, rate )
13 {
14    setBaseSalary( salary ); // validate and store base salary
15
16    cout << "BasePlusCommissionEmployee constructor: " << endl;
17    print();
18    cout << "\n\n";
19 } // end BasePlusCommissionEmployee constructor
20
```

Fig. 12.25 | `BasePlusCommissionEmployee`'s constructor and destructor output text. (Part 1 of 3.)

```
21  // destructor
22  BasePlusCommissionEmployee::~BasePlusCommissionEmployee()
23  {
24     cout << "BasePlusCommissionEmployee destructor: " << endl;
25     print();
26     cout << "\n\n";
27  } // end BasePlusCommissionEmployee destructor
28
29  // set base salary
30  void BasePlusCommissionEmployee::setBaseSalary( double salary )
31  {
32     baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
33  } // end function setBaseSalary
34
35  // return base salary
36  double BasePlusCommissionEmployee::getBaseSalary() const
37  {
38     return baseSalary;
39  } // end function getBaseSalary
40
```

Fig. 12.25 | `BasePlusCommissionEmployee`'s constructor and destructor output text. (Part 2 of 3.)

```
41   // calculate earnings
42   double BasePlusCommissionEmployee::earnings() const
43   {
44      return getBaseSalary() + CommissionEmployee::earnings();
45   } // end function earnings
46
47   // print BasePlusCommissionEmployee object
48   void BasePlusCommissionEmployee::print() const
49   {
50      cout << "base-salaried ";
51
52      // invoke CommissionEmployee's print function
53      CommissionEmployee::print();
54
55      cout << "\nbase salary: " << getBaseSalary();
56   } // end function print
```

Fig. 12.25 | `BasePlusCommissionEmployee`'s constructor and destructor output text. (Part 3 of 3.)

```
1  // Fig. 12.26: fig12_26.cpp
2  // Display order in which base-class and derived-class constructors
3  // and destructors are called.
4  #include <iostream>
5  #include <iomanip>
6  #include "BasePlusCommissionEmployee.h"
7  using namespace std;
8
9  int main()
10 {
11    // set floating-point output formatting
12    cout << fixed << setprecision( 2 );
13
14    { // begin new scope
15       CommissionEmployee employee1(
16          "Bob", "Lewis", "333-33-3333", 5000, .04 );
17    } // end scope
18
19    cout << endl;
20    BasePlusCommissionEmployee
21       employee2( "Lisa", "Jones", "555-55-5555", 2000, .06, 800 );
22
```

Fig. 12.26 | Constructor and destructor call order. (Part 1 of 4.)

```
23      cout << endl;
24      BasePlusCommissionEmployee
25         employee3( "Mark", "Sands", "888-88-8888", 8000, .15, 2000 );
26      cout << endl;
27   } // end main
```

```
CommissionEmployee constructor:
commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04

CommissionEmployee destructor:
commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04


CommissionEmployee constructor:
commission employee: Lisa Jones
social security number: 555-55-5555
gross sales: 2000.00
commission rate: 0.06
```

Fig. 12.26 | Constructor and destructor call order. (Part 2 of 4.)

```
BasePlusCommissionEmployee constructor:
base-salaried commission employee: Lisa Jones
social security number: 555-55-5555
gross sales: 2000.00
commission rate: 0.06
base salary: 800.00


CommissionEmployee constructor:
commission employee: Mark Sands
social security number: 888-88-8888
gross sales: 8000.00
commission rate: 0.15

BasePlusCommissionEmployee constructor:
base-salaried commission employee: Mark Sands
social security number: 888-88-8888
gross sales: 8000.00
commission rate: 0.15
base salary: 2000.00
```

2

3

4

Fig. 12.26 | Constructor and destructor call order. (Part 3 of 4.)

```
BasePlusCommissionEmployee destructor:
base-salaried commission employee: Mark Sands
social security number: 888-88-8888
gross sales: 8000.00
commission rate: 0.15
base salary: 2000.00

CommissionEmployee destructor:
commission employee: Mark Sands
social security number: 888-88-8888
gross sales: 8000.00
commission rate: 0.15

BasePlusCommissionEmployee destructor:
base-salaried commission employee: Lisa Jones
social security number: 555-55-5555
gross sales: 2000.00
commission rate: 0.06
base salary: 800.00

CommissionEmployee destructor:
commission employee: Lisa Jones
social security number: 555-55-5555
gross sales: 2000.00
commission rate: 0.06
```

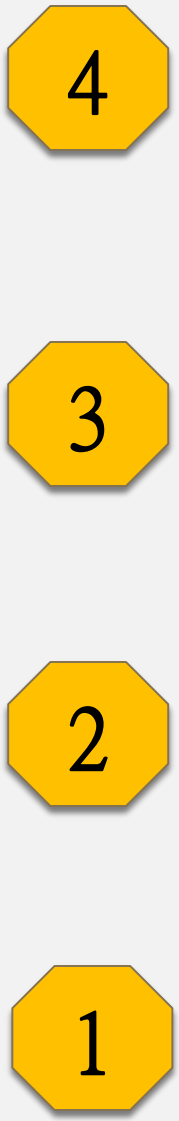


Fig. 12.26 | Constructor and destructor call order. (Part 4 of 4.)

Section 12.4.3 (error case)

Derived class cannot access private member of base class

- Strict protection on private data member
  - Even derived class cannot access private members
- Error example in Section 12.4.3
- How to change it?
  - (1) Use protected data member in base class OR
    - Example in Section 12.4.4
  - (2) Use Get function (declared public) in base class
    - Example in Section 12.4.5

Summary: Chapter 12

- Inheritance
  - class DerivedClassName : inheritanceType BaseClassName

- Protected member
- Derived class cannot access private data member in base class