BİL-142 Bilgisayar Programlama II (C/C++)

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- **`protected`** Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- **`public`**, **`protected`** ve **`private`** Inheritance

Giriş

- Inheritance (miras alma), bir sınıfa ait data ve fonksiyonların tekrar kullanılarak daha gelişmiş sınıf üretilmesini sağlar.
- Tüm data üyeleri ve fonksiyon üyelerini yeniden oluşturarak bir yeni sınıf oluşturmak yerine mevcut bir sınıfın üyelerini kullanarak yeni bir sınıf üretmek daha kolaydır.
- Mevcut sınıf **base class**, türetilen sınıf ise **derived class** olarak adlandırılır.
- Türetilmiş bir class, base class'ın private üyelerine doğrudan erişemez.

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- `protected` Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- `public`, `protected` ve `private` Inheritance

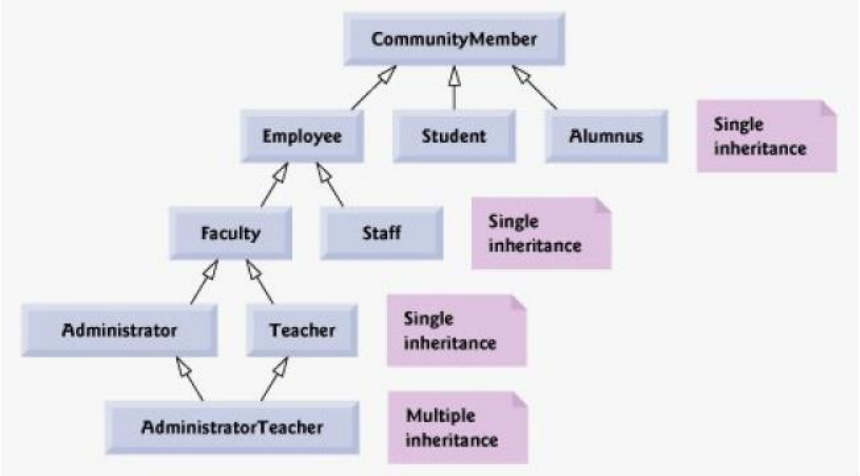
Temel Sınıflar ve Türetilmiş Sınıflar

- Bir sınıftan başka sınıflar üretilebilir. Bir çizgi class'ı kullanılarak bir dikdörtgen veya başka şekil için yeni bir class türetilebilir.
- Türetilen yeni sınıf (derived-class), türetildiği sınıfın (base class) bazı öelliklerini kullanabilir.
- Araçlar base class olarak tanımlanırsa, çok sayıda araç (otomobil, traktör, bot, uçak, bisiklet, ...) üretilebilir.
- Bir üniversitede çalışanlar için base class oluşturulursa, akademik ve idari çalışanlara yönelik iki sınıf türetilebilir.
- Bir sınıf birden fazla sınıf kullanılarak türetilebilir.

5

Temel Sınıflar ve Türetilmiş Sınıflar

- Bir sınıftan (single inheritance) ve birden fazla sınıftan (multiple inheritance) türetilen sınıflar.



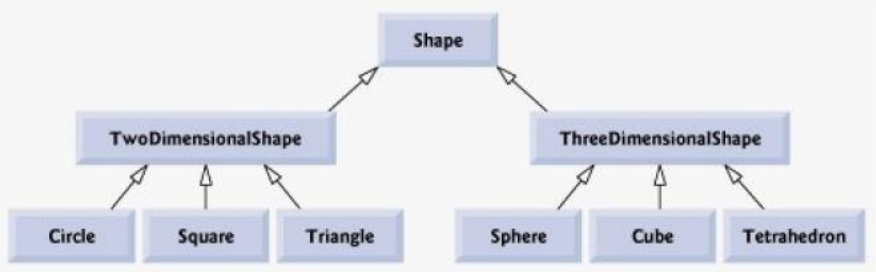
6

Temel Sınıflar ve Türetilmiş Sınıflar

- Şekilde Shape inheritance hiyerarşisi görülmektedir.
- Aşağıdaki kod **public inheritance** yapmaktadır.

```
class TwoDimensionalShape : public Shape
```

- Tüm inheritance yapılarında (private, protected, public), base class içindeki private üyelere erişlemez ancak türetilmiş sınıfında üyeleri sayılırlar.



7

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- **`protected`** Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- **`public`**, **`protected`** ve **`private`** Inheritance

`protected` Üyeler

- Bir base class'ın public üyelerine, class üyeleri, oluşturulan nesne, türetilmiş class'lardan oluşturulan nesneler tarafından erişilebilir.
- Bir base class'ın private üyelerine sadece o class içindeki fonksiyonlar ve friends fonksiyonlar tarafından erişilebilir.
- **`protected`** ise **`public`** ve **`private`** arasında erişim sağlar.
- Bir base class'ın protected üyelerine o class'ın üyeleri, friends fonksiyonları, o class'tan türetilen class'ların üyeleri ve friends fonksiyonları tarafından erişilebilir.

9

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- **`protected`** Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- **`public`**, **`protected`** ve **`private`** Inheritance

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- Bir şirketin çalışanlarına yönelik uygulamada, komisyonla çalışan elemanlar (commission employees) ve sabit ücret+komisyonla çalışan elemanlar (base plus commission employees) olsun.
- Bu şirket için commission employees ile base salaried commission employees arasındaki ilişki aşağıdaki farklı örneklerle oluşturulabilir:
  - İlk örnekte, **`CommissionEmployee`** class'ı oluşturulup, **`first name`**, **`last name`**, **`social security number`**, **`commission rate`** ve **`gross sales amount`** değişkenleri private tanımlanabilir.
  - İkinci örnekte, **`BasePlusCommissionEmployee`** class'ı oluşturulup, **`first name`**, **`last name`**, **`social security number`**, **`commission rate`**,**`gross sales amount`** ve **`base salary`** değişkenleri private tanımlanır.

11

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- Üçüncü örnekte, **`BasePlusCommissionEmployee`** class'ının **`CommissionEmployee`** class'ından yeni bir versiyonu oluşturulur ve private üyelere ulaşılmaya çalışılır. Bu durumda compile hatası oluşur çünkü türetilmiş class base class'ın private üyelerine ulaşamaz.
- Dördüncü örnekte, **`CommissionEmployee`** class'ının dataları protected tanımlanır, yeni bir **`BasePlusCommissionEmployee`** class'ı oluşturulur ve **`CommissionEmployee`** class'ının datalarına doğrudan ulaşılabilir.
- Beşinci örnekte, **`CommissionEmployee`** class'ının dataları private tanımlanır, **`BasePlusCommissionEmployee`** class'ı türetilir ve **`CommissionEmployee`** class'ının fonksiyonlarıyla private datalara erişilir.

12

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`CommisionEmployee`** class'ının oluşturulması

```
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
26 // return first name
27 string CommissionEmployee::getFirstName() const
28 {
29    return firstName;
30 } // end function getFirstName
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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
74  // return commission rate
75  double CommissionEmployee::getCommissionRate() const
76  {
77     return commissionRate;
78  } // end function getCommissionRate
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 << ' ' << lastName
90        << "\nsocial security number: " << socialSecurityNumber
91        << "\ngross sales: " << grossSales
92        << "\ncommission rate: " << commissionRate;
93  } // end function print
94
```

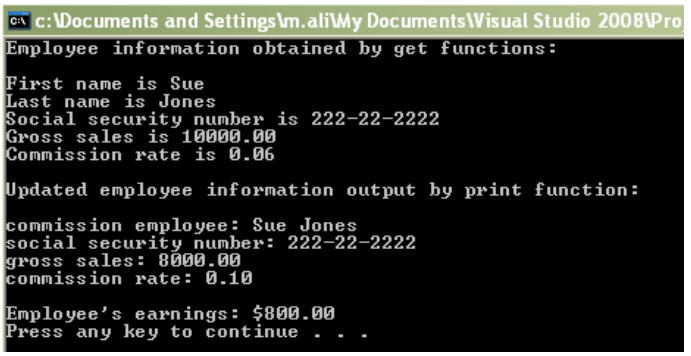
Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
// Fig. 12.6: fig12_06.cpp
// Testing class CommissionEmployee.
#include <iostream>
using std::cout;
using std::endl;
using std::fixed;

#include <iomanip>
using std::setprecision;

#include "CommissionEmployee.h" // CommissionEmployee class definition

int main()
{
   // instantiate a CommissionEmployee object
   CommissionEmployee employee(
      "Sue", "Jones", "222-22-2222", 10000, .06 );

   // set floating-point output formatting
   cout << fixed << setprecision( 2 );

```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
   // get commission employee data
   cout << "Employee information obtained by get functions: \n"
      << "\nFirst name is " << employee.getFirstName()
      << "\nLast name is " << employee.getLastName()
      << "\nSocial security number is "
      << employee.getSocialSecurityNumber()
      << "\nGross sales is " << employee.getGrossSales()
      << "\nCommission rate is " << employee.getCommissionRate() << endl;

   employee.setGrossSales( 8000 ); // set gross sales
   employee.setCommissionRate( .1 ); // set commission rate

   cout << "\nUpdated employee information output by print function: \n"
      << endl;
   employee.print(); // display the new employee information

   // display the employee's earnings
   cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;

   return 0;
} // end main
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki



Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`CommissionEmployee`** class'ı, bir **`constructor`**, **`earnings`** isimli fonksiyon ve **`print`** isimli fonksiyon üyelerine sahiptir.
- **`get`** ve **`set`** fonksiyonları **`firstname`**, **`lastname`**, **`socialSecurityNumber`**, **`grossSales`** ve **`commisionRate`** dataları üzerinde işlem yapmak için kullanılır.
- **`CommissionEmployee`** class'ının constructor'ı data üyelerine ilk değerlerini atar.

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`BasePlusCommissionEmployee`** class'ının inheritance olmadan oluşturulması

```
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 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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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    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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
22 // set first name
23 void BasePlusCommissionEmployee::setFirstName( const string &first )
24 {
25    firstName = first; // should validate
26 } // end function setFirstName
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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
46  // set social security number
47  void BasePlusCommissionEmployee::setSocialSecurityNumber(
48     const string &ssn )
49  {
50     socialSecurityNumber = ssn; // should validate
51  } // end function setSocialSecurityNumber
52
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
110
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.9: fig12_09.cpp
2  // Testing class BasePlusCommissionEmployee.
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 // BasePlusCommissionEmployee class definition
12 #include "BasePlusCommissionEmployee.h"
13
14 int main()
15 {
16    // instantiate BasePlusCommissionEmployee object
17    BasePlusCommissionEmployee
18       employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
19
20    // set floating-point output formatting
21    cout << fixed << setprecision( 2 );
22
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
23    // get commission employee data
24    cout << "Employee information obtained by get functions: \n"
25       << "\nFirst name is " << employee.getFirstName()
26       << "\nLast name is " << employee.getLastName()
27       << "\nSocial security number is "
28       << employee.getSocialSecurityNumber()
29       << "\nGross sales is " << employee.getGrossSales()
30       << "\nCommission rate is " << employee.getCommissionRate()
31       << "\nBase salary is " << employee.getBaseSalary() << endl;
32
33    employee.setBaseSalary( 1000 ); // set base salary
34
35    cout << "\nUpdated employee information output by print function: \n"
36       << endl;
37    employee.print(); // display the new employee information
38
39    // display the employee's earnings
40    cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
41
42    return 0;
43 } // end main
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
c:\Documents and Settings\m.ali\My Documents\Visual Studio 2008\P
Employee information obtained by get functions:

First name is Bob
Last name is Lewis
Social security number is 333-33-3333
Gross sales is 5000.00
Commission rate is 0.04
Base salary is 300.00

Updated employee information output by print function:

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 1000.00

Employee's earnings: $1200.00
Press any key to continue . . .
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`CommissionEmployee`** class'ı ile **`BasePlusCommissionEmployee`** class'ının birçok kodu aynıdır. Örneğin **`firstName`**, **`lastName`**, **`getFirstName`**, **`setFirstName`**, **`setLastName`** ve **`getLastName`** aynıdır.
- **`get`** ve **`set`** fonksiyonları ile ikisi de private data üyeleri **`socialSecurityNumber`**, **`grossSales`** ve **`commisionRate`** üzerinde işlem yapıyorlar.
- **`BasePlusCommissionEmployee`** class'ının constructor'ı farklı olarak **`baseSalary`** değerini set ediyor.
- **`BasePlusCommissionEmployee`** class'ının diğer farklılıkları **`setBaseSalary`** ve **`getBaseSalary`** fonksiyonlarıdır.

33

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`BasePlusCommissionEmployee`** class'ının inheritance kullanılarak oluşturulması

```
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
```

34

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
25 // return base salary
26 double BasePlusCommissionEmployee::getBaseSalary() const
27 {
28    return baseSalary;
29 } // end function getBaseSalary
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
```

Türetilen class, base class'ın private üyelerine erişemez.

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`BasePlusCommissionEmployee`** class'ı **`CommissionEmployee`** class'ı kullanılarak türetilmiştir.
- **`public`** anahtar kelimesi, **`BasePlusCommissionEmployee`** class'ının **`CommissionEmployee`** class'ının tüm üyelerini alacağını gösterir.
- Türetilmiş class'lar base class'ların constructor'larını alamazlar. Her class kendi constructor'ına sahiptir.
- Önceki örnekte **`BasePlusCommissionEmployee`** class'ı, **`CommissionEmployee`** class'ının private üyelerine erişmeye çalıştığı için compile hatası verir.
- Satır 35'te, **`getCommisionRate`** ve **`getGrossSales`** fonksiyonları ile aynı değerler alınabilir. Aynı değişiklik Satır 42-45'tede yapılabilir.

37

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`protected`** data kullanarak **`BasePlusCommissionEmployee-CommissionEmployee`** inheritance oluşturulması

```
1  // Fig. 12.12: CommissionEmployee.h
2  // CommissionEmployee class definition with protected data.
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
```

38

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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 protected:
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
```

39

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.13: 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
```

40

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
26  // return first name
27  string CommissionEmployee::getFirstName() const
28  {
29     return firstName;
30  } // end function getFirstName
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
```

41

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
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
```

42

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
74  // return commission rate
75  double CommissionEmployee::getCommissionRate() const
76  {
77     return commissionRate;
78  } // end function getCommissionRate
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 << ' ' << lastName
90        << "\nsocial security number: " << socialSecurityNumber
91        << "\ngross sales: " << grossSales
92        << "\ncommission rate: " << commissionRate;
93  } // end function print
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.14: 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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.15: 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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
30
31 // calculate earnings
32 double BasePlusCommissionEmployee::earnings() const
33 {
34    // can access protected data of base class
35    return baseSalary + ( commissionRate * grossSales );
36 } // end function earnings
37
38 // print BasePlusCommissionEmployee object
39 void BasePlusCommissionEmployee::print() const
40 {
41    // can access protected data of base class
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
```

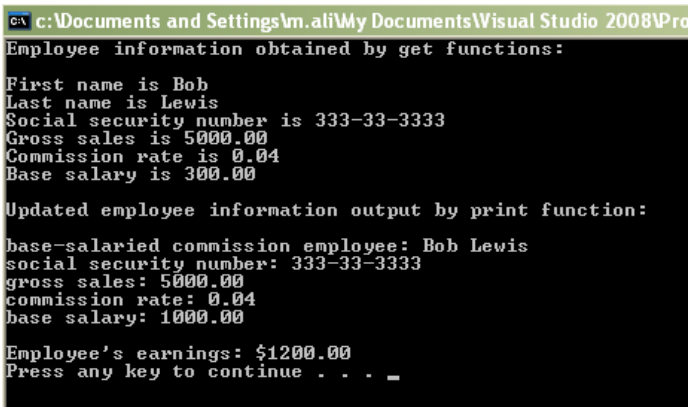
Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.16: fig12_16.cpp
2  // Testing class BasePlusCommissionEmployee.
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 // BasePlusCommissionEmployee class definition
12 #include "BasePlusCommissionEmployee.h"
13
14 int main()
15 {
16    // instantiate BasePlusCommissionEmployee object
17    BasePlusCommissionEmployee
18       employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
19
20    // set floating-point output formatting
21    cout << fixed << setprecision( 2 );
22
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
23    // get commission employee data
24    cout << "Employee information obtained by get functions: \n"
25       << "\nFirst name is " << employee.getFirstName()
26       << "\nLast name is " << employee.getLastName()
27       << "\nSocial security number is "
28       << employee.getSocialSecurityNumber()
29       << "\nGross sales is " << employee.getGrossSales()
30       << "\nCommission rate is " << employee.getCommissionRate()
31       << "\nBase salary is " << employee.getBaseSalary() << endl;
32
33    employee.setBaseSalary( 1000 ); // set base salary
34
35    cout << "\nUpdated employee information output by print function: \n"
36       << endl;
37    employee.print(); // display the new employee information
38
39    // display the employee's earnings
40    cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
41
42    return 0;
43 } // end main
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki



Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`BasePlusCommissionEmployee`** class'ının **`CommissionEmployee`** class'ındaki **`firstName`**, **`lastName`**, **`socialSecurityNumber`**, **`grossSales`** ve **`commissionRate`** data üylerine ulaşabilmesi için **`protected`** tanımlanması gereklidir.
- **`protected`** data üyeleri bir class'ın tüm üyeleri, friends fonksiyonları ve o class'tan türetilmiş tüm class'ların üyeleri ve friends fonksiyonları tarafından erişilebilir.

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- Önceki örnekte **`CommissionEmployee`** class'ında **`firstName`**, **`lastName`**, **`socialSecurityNumber`**, **`grossSales`** ve **`commissionRate`** data üyleri **`protected`** tanımlanmıştır.
- **`protected`** tanımlanan data üyelerine **`get`** ve **`set`** fonksiyonları olmadan erişilebildiği için performans artar.
- Ancak **`protected`** tanımlanan data üyelerine türetilmiş class'lar tarafından geçersiz değer atanabilir.

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`private`** data kullanarak **`BasePlusCommissionEmployee-CommissionEmployee`** inheritance oluşturulması

```
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 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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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
```

53

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.18: 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    : firstName( first ), lastName( last ), socialSecurityNumber( ssn )
13 {
14    setGrossSales( sales ); // validate and store gross sales
15    setCommissionRate( rate ); // validate and store commission rate
16 } // end CommissionEmployee constructor
17
18 // set first name
19 void CommissionEmployee::setFirstName( const string &first )
20 {
21    firstName = first; // should validate
22 } // end function setFirstName
23
```

54

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
24  // return first name
25  string CommissionEmployee::getFirstName() const
26  {
27     return firstName;
28  } // end function getFirstName
29
30  // set last name
31  void CommissionEmployee::setLastName( const string &last )
32  {
33     lastName = last; // should validate
34  } // end function setLastName
35
36  // return last name
37  string CommissionEmployee::getLastName() const
38  {
39     return lastName;
40  } // end function getLastName
41
42  // set social security number
43  void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
44  {
45     socialSecurityNumber = ssn; // should validate
46  } // end function setSocialSecurityNumber
47
```

55

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
48  // return social security number
49  string CommissionEmployee::getSocialSecurityNumber() const
50  {
51     return socialSecurityNumber;
52  } // end function getSocialSecurityNumber
53
54  // set gross sales amount
55  void CommissionEmployee::setGrossSales( double sales )
56  {
57     grossSales = ( sales < 0.0 ) ? 0.0 : sales;
58  } // end function setGrossSales
59
60  // return gross sales amount
61  double CommissionEmployee::getGrossSales() const
62  {
63     return grossSales;
64  } // end function getGrossSales
65
66  // set commission rate
67  void CommissionEmployee::setCommissionRate( double rate )
68  {
69     commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
70  } // end function setCommissionRate
71
72  // return commission rate
73  double CommissionEmployee::getCommissionRate() const
74  {
75     return commissionRate;
76  } // end function getCommissionRate
77
```

56

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

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


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
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  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
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.20: 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
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

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

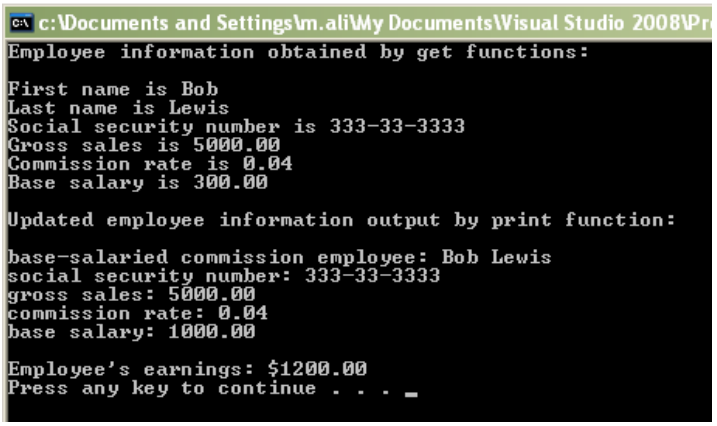
Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
1  // Fig. 12.21: fig12_21.cpp
2  // Testing class BasePlusCommissionEmployee.
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 // BasePlusCommissionEmployee class definition
12 #include "BasePlusCommissionEmployee.h"
13
14 int main()
15 {
16    // instantiate BasePlusCommissionEmployee object
17    BasePlusCommissionEmployee
18       employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
19
20    // set floating-point output formatting
21    cout << fixed << setprecision( 2 );
22
```


Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

```
23    // get commission employee data
24    cout << "Employee information obtained by get functions: \n"
25       << "\nFirst name is " << employee.getFirstName()
26       << "\nLast name is " << employee.getLastName()
27       << "\nSocial security number is "
28       << employee.getSocialSecurityNumber()
29       << "\nGross sales is " << employee.getGrossSales()
30       << "\nCommission rate is " << employee.getCommissionRate()
31       << "\nBase salary is " << employee.getBaseSalary() << endl;
32
33    employee.setBaseSalary( 1000 ); // set base salary
34
35    cout << "\nUpdated employee information output by print function: \n"
36       << endl;
37    employee.print(); // display the new employee information
38
39    // display the employee's earnings
40    cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
41
42    return 0;
43 } // end main
```

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki



Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- En iyi yazılım mühendisliği yaklaşımı base class data üyelerinin **`private`** tanımlanması ve **`public`** fonksiyonlara erişime izin verilmesidir. Önceki örnekte bu yönde yeniden düzenleme yapılmıştır.
- **`CommissionEmployee`** class'ının data üyeleri **`firstName`**, **`lastName`**, **`socialSecurityNumber`**, **`grossSales`** ve **`commissionRate`** **`private`** tanımlanmış ve **`public`** fonksiyonlar olan **`getFirstName`**, **`setFirstName`**, **`getLastName`**, **`setLastName`**, **`getSocialSecurityNumber`**, **`setSocialSecurityNumber`**, **`getGrossSales`**, **`setGrossSales`**, **`setCommissionRate`**, **`getCommissionRate`**, **`earnings`** ve **`print`** ile erişilebilmektedir.

Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki

- **`BasePlusCommissionEmployee`** class'ı earnings fonksiyonuna sahiptir. Ancak **`CommissionEmployee`** class'ının **`earnings`** fonksiyonu ile **`baseSalary`** değerini toplamaktadır.
- **`CommissionEmployee`** class'ının fonksiyonlarına türetilmiş class içinden erişmek için **`::`** operatörü kullanılır (**`CommissionEmployee::earnings()`**).
- Türetilmiş class içinden base class fonksiyonunu çağırırken **`::`** kullanılmazsa sonsuz recursion oluşur.

65

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- **`protected`** Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- **`public`**, **`protected`** ve **`private`** Inheritance

Türetilmiş Sınıflarda Constructor ve Destructor

- Bir türetilmiş class'tan nesne üretildiğinde, önce kendi constructor'ını çağırır. Kendi constructor'ı çalıştırılmadan önce base class constructor'ı çalıştırılır.
- Eğer base class'ta başka class'tan türetilmişse bu işlem önceki class içinde yapılır.
- Önceki örneklerde **`BasePlusCommissionEmployee`** class'ından bir nesne oluşturulduğunda, **`CommissionEmployee`** constructor'ı çağırılır.
- **`CommissionEmployee`** constructor'ının çalışması bitince **`BasePlusCommissionEmployee`** constructor'ı çağırılır.
- Bir türetilmiş class'ın destructor'ı çağırıldığında, sırasıyla kendi destructor'ı ve base class destructor'ı çalıştırılır. Bu işlem hiyerarşik olarak son base class'a kadar kaskad devam eder.

67

Türetilmiş Sınıflarda Constructor ve Destructor

```
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 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    ~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
24
```

68

Türetilmiş Sınıflarda Constructor ve Destructor

```
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
```


Türetilmiş Sınıflarda Constructor ve Destructor

```
1  // Fig. 12.23: CommissionEmployee.cpp
2  // Class CommissionEmployee member-function definitions.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  #include "CommissionEmployee.h" // CommissionEmployee class definition
8
9  // constructor
10 CommissionEmployee::CommissionEmployee(
11    const string &first, const string &last, const string &ssn,
12    double sales, double rate )
13    : firstName( first ), lastName( last ), socialSecurityNumber( ssn )
14 {
15    setGrossSales( sales ); // validate and store gross sales
16    setCommissionRate( rate ); // validate and store commission rate
17
18    cout << "CommissionEmployee constructor: " << endl;
19    print();
20    cout << "\n\n";
21 } // end CommissionEmployee constructor
22
```

Türetilmiş Sınıflarda Constructor ve Destructor

```
23  // destructor
24  CommissionEmployee::~CommissionEmployee()
25  {
26     cout << "CommissionEmployee destructor: " << endl;
27     print();
28     cout << "\n\n";
29  } // end CommissionEmployee destructor
30
31  // set first name
32  void CommissionEmployee::setFirstName( const string &first )
33  {
34     firstName = first; // should validate
35  } // end function setFirstName
36
37  // return first name
38  string CommissionEmployee::getFirstName() const
39  {
40     return firstName;
41  } // end function getFirstName
42
43  // set last name
44  void CommissionEmployee::setLastName( const string &last )
45  {
46     lastName = last; // should validate
47  } // end function setLastName
48
```

71

Türetilmiş Sınıflarda Constructor ve Destructor

```
49  // return last name
50  string CommissionEmployee::getLastName() const
51  {
52     return lastName;
53  } // end function getLastName
54
55  // set social security number
56  void CommissionEmployee::setSocialSecurityNumber( const string &ssn )
57  {
58     socialSecurityNumber = ssn; // should validate
59  } // end function setSocialSecurityNumber
60
61  // return social security number
62  string CommissionEmployee::getSocialSecurityNumber() const
63  {
64     return socialSecurityNumber;
65  } // end function getSocialSecurityNumber
66
67  // set gross sales amount
68  void CommissionEmployee::setGrossSales( double sales )
69  {
70     grossSales = ( sales < 0.0 ) ? 0.0 : sales;
71  } // end function setGrossSales
72
```

72

Türetilmiş Sınıflarda Constructor ve Destructor

```
// return gross sales amount
double CommissionEmployee::getGrossSales() const
{
   return grossSales;
} // end function getGrossSales

// set commission rate
void CommissionEmployee::setCommissionRate( double rate )
{
   commissionRate = ( rate > 0.0 && rate < 1.0 ) ? rate : 0.0;
} // end function setCommissionRate

// return commission rate
double CommissionEmployee::getCommissionRate() const
{
   return commissionRate;
} // end function getCommissionRate

// calculate earnings
double CommissionEmployee::earnings() const
{
   return getCommissionRate() * getGrossSales();
} // end function earnings

// print CommissionEmployee object
void CommissionEmployee::print() const
{
   cout << "commission employee: "
      << getFirstName() << ' ' << getLastName()
      << "\nsocial security number: " << getSocialSecurityNumber()
      << "\ngross sales: " << getGrossSales()
      << "\ncommission rate: " << getCommissionRate();
} // end function print
```


Türetilmiş Sınıflarda Constructor ve Destructor

```
// Fig. 12.24: BasePlusCommissionEmployee.h
// BasePlusCommissionEmployee class derived from class
// CommissionEmployee.
#ifndef BASEPLUS_H
#define BASEPLUS_H

#include <string> // C++ standard string class
using std::string;

#include "CommissionEmployee.h" // CommissionEmployee class declaration

class BasePlusCommissionEmployee : public CommissionEmployee
{
public:
   BasePlusCommissionEmployee( const string &, const string &,
      const string &, double = 0.0, double = 0.0, double = 0.0 );
   ~BasePlusCommissionEmployee(); // destructor

   void setBaseSalary( double ); // set base salary
   double getBaseSalary() const; // return base salary

   double earnings() const; // calculate earnings
   void print() const; // print BasePlusCommissionEmployee object
private:
   double baseSalary; // base salary
}; // end class BasePlusCommissionEmployee

#endif
```

Türetilmiş Sınıflarda Constructor ve Destructor

```
1  // Fig. 12.25: BasePlusCommissionEmployee.cpp
2  // Class BasePlusCommissionEmployee member-function definitions.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  // BasePlusCommissionEmployee class definition
8  #include "BasePlusCommissionEmployee.h"
9
10 // constructor
11 BasePlusCommissionEmployee::BasePlusCommissionEmployee(
12    const string &first, const string &last, const string &ssn,
13    double sales, double rate, double salary )
14    // explicitly call base-class constructor
15    : CommissionEmployee( first, last, ssn, sales, rate )
16 {
17    setBaseSalary( salary ); // validate and store base salary
18
19    cout << "BasePlusCommissionEmployee constructor: " << endl;
20    print();
21    cout << "\n\n";
22 } // end BasePlusCommissionEmployee constructor
23
24 // destructor
25 BasePlusCommissionEmployee::~BasePlusCommissionEmployee()
26 {
27    cout << "BasePlusCommissionEmployee destructor: " << endl;
28    print();
29    cout << "\n\n";
30 } // end BasePlusCommissionEmployee destructor
31
```


Türetilmiş Sınıflarda Constructor ve Destructor

```
32 // set base salary
33 void BasePlusCommissionEmployee::setBaseSalary( double salary )
34 {
35    baseSalary = ( salary < 0.0 ) ? 0.0 : salary;
36 } // end function setBaseSalary
37
38 // return base salary
39 double BasePlusCommissionEmployee::getBaseSalary() const
40 {
41    return baseSalary;
42 } // end function getBaseSalary
43
44 // calculate earnings
45 double BasePlusCommissionEmployee::earnings() const
46 {
47    return getBaseSalary() + CommissionEmployee::earnings();
48 } // end function earnings
49
50 // print BasePlusCommissionEmployee object
51 void BasePlusCommissionEmployee::print() const
52 {
53    cout << "base-salaried ";
54
55    // invoke CommissionEmployee's print function
56    CommissionEmployee::print();
57
58    cout << "\nbase salary: " << getBaseSalary();
59 } // end function print
```

Türetilmiş Sınıflarda Constructor ve Destructor

```
// Fig. 12.26: fig12_26.cpp
// Display order in which base-class and derived-class constructors
// and destructors are called.
#include <iostream>
using std::cout;
using std::endl;
using std::fixed;

#include <iomanip>
using std::setprecision;

// BasePlusCommissionEmployee class definition
#include "BasePlusCommissionEmployee.h"

int main()
{
   // set floating-point output formatting
   cout << fixed << setprecision( 2 );

   { // begin new scope
      CommissionEmployee employee1(
         "Bob", "Lewis", "333-33-3333", 5000, .04 );
   } // end scope

   cout << endl;
   BasePlusCommissionEmployee
      employee2( "Lisa", "Jones", "555-55-5555", 2000, .06, 800 );

   cout << endl;
   BasePlusCommissionEmployee
      employee3( "Mark", "Sands", "888-88-8888", 8000, .15, 2000 );
   cout << endl;
   return 0;
} // end main
```


Türetilmiş Sınıflarda Constructor ve Destructor

```
c:\Documents and Settings\m.ali\My Documents\Visual Studio 2
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

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


Press any key to continue . . .
```

Konular

- Giriş
- Temel (Base) Sınıflar ve Türetilmiş (Derived) Sınıflar
- **`protected`** Üyeler
- Temel Sınıflar ve Türetilmiş Sınıflar Arasındaki İlişki
- Türetilmiş Sınıflarda Constructor ve Destructor
- **`public`**, **`protected`** ve **`private`** Inheritance

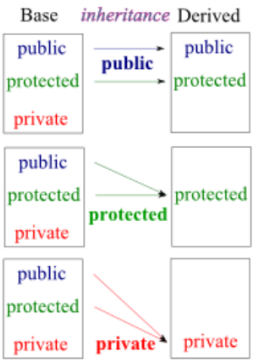
`public`, **`protected`** ve **`private`** Inheritance

- Bir class kullanılarak yeni bir class türetilirken **`public`**, **`private`** veya **`protected`** inheritance yapılabilir.
- Genellikle **`public`** inheritance kullanılır.
- Bir base class'tan yeni bir class **`public`** olarak türetilirse, base class'ın **`public`** üyeleri türetilmiş class'ın **`public`** üyeleri ve base class'ın **`protected`** üyeleri türetilmiş class'ın **`protected`** üyeleri olur.
- Türetilmiş class ile base class'ın private üyelerine ulaşılamaz.
- Bir base class'tan yeni bir class **`protected`** olarak türetilirse, base class'ın **`public ve protected`** üyeleri türetilmiş class'ın **`protected`** üyeleri olur.
- Bir base class'tan yeni bir class **`private`** olarak türetilirse, base class'ın **`public ve protected`** üyeleri türetilmiş class'ın **`private`** üyeleri olur.

80

`public`, `protected` ve `private` Inheritance

- Türetilmiş sınıftaki erişim belrileyicisi base sınıftakinden daha geniş olamaz.



`public`, `protected` ve `private` Inheritance

Base-class member-access specifier	Type of inheritance		
	`public` inheritance	`protected` inheritance	`private` inheritance
`public`	`public` in derived class. Can be accessed directly by member functions, `friend` functions and nonmember functions.	`protected` in derived class. Can be accessed directly by member functions and `friend` functions.	`private` in derived class. Can be accessed directly by member functions and `friend` functions.
`protected`	`protected` in derived class. Can be accessed directly by member functions and `friend` functions.	`protected` in derived class. Can be accessed directly by member functions and `friend` functions.	`private` in derived class. Can be accessed directly by member functions and `friend` functions.
`private`	Hidden in derived class. Can be accessed by member functions and `friend` functions through `public` or `protected` member functions of the base class.	Hidden in derived class. Can be accessed by member functions and `friend` functions through `public` or `protected` member functions of the base class.	Hidden in derived class. Can be accessed by member functions and `friend` functions through `public` or `protected` member functions of the base class.

`public`, `protected` ve `private` Inheritance

- Türetilen sınıfta kalıtım şekline göre erişim yapılabilir.

```
class A
{
    public:
        int x;
    protected:
        int y;
    private:
        int z;
};

class B : public A
{
    // x is public
    // y is protected
    // z is not accessible from B
};

class C : protected A
{
    // x is protected
    // y is protected
    // z is not accessible from C
};

class D : private A    // 'private' is default for classes
{
    // x is private
    // y is private
    // z is not accessible from D
};
```


`public`, `protected` ve `private` Inheritance

- `public` kalıtım

```
class Base
{
public:
    int m_public;
protected:
    int m_protected;
private:
    int m_private;
};

class Pub: public Base // note: public inheritance
{
    // Public inheritance means:
    // Public inherited members stay public (so m_public is treated as public)
    // Protected inherited members stay protected (so m_protected is treated as protected)
    // Private inherited members stay inaccessible (so m_private is inaccessible)
public:
    Pub()
    {
        m_public = 1;       // okay: m_public was inherited as public
        m_protected = 2;    // okay: m_protected was inherited as protected
        m_private = 3;      // not okay: m_private is inaccessible from derived class
    }
};

int main()
{
    // Outside access uses the access specifiers of the class being accessed.
    Base base;
    base.m_public = 1;      // okay: m_public is public in Base
    base.m_protected = 2;   // not okay: m_protected is protected in Base
    base.m_private = 3;     // not okay: m_private is private in Base

    Pub pub;
    pub.m_public = 1;       // okay: m_public is public in Pub
    pub.m_protected = 2;    // not okay: m_protected is protected in Pub
    pub.m_private = 3;      // not okay: m_private is inaccessible in Pub
    return 0;
}
```

`public`, `protected` ve `private` Inheritance

- `protected` kalıtım

```
class Base
{
public:
    int m_public;
protected:
    int m_protected;
private:
    int m_private;
};

class Pro: protected Base // note: protected inheritance
{
public:
    Pro()
    {
        m_public = 1;    // okay: m_public is now protected in Pro
        m_protected = 2; // okay: m_protected is now protected in Pro
        m_private = 3;   // not okay: derived classes can't access private members in the base class
    }
};

int main()
{
    // Outside access uses the access specifiers of the class being accessed.
    // In this case, the access specifiers of base.
    Base base;
    base.m_public = 1;       // okay: m_public is public in Base
    base.m_protected = 2;    // not okay: m_protected is protected in Base
    base.m_private = 3;      // not okay: m_private is private in Base

    Pro pro;
    pro.m_public = 1;        // not okay: m_public is now protected in Pro
    pro.m_protected = 2;     // not okay: m_protected is protected in Pro
    pro.m_private = 3;       // not okay: m_private is inaccessible in Pro

    return 0;
}
```

85

`public`, `protected` ve `private` Inheritance

- `private` kalıtım

```
class Base
{
public:
    int m_public;
protected:
    int m_protected;
private:
    int m_private;
};

class Pri: private Base // note: private inheritance
{
    // Private inheritance means:
    // Public inherited members become private (so m_public is treated as private)
    // Protected inherited members become private (so m_protected is treated as private)
    // Private inherited members stay inaccessible (so m_private is inaccessible)
public:
    Pri()
    {
        m_public = 1;       // okay: m_public is now private in Pri
        m_protected = 2;    // okay: m_protected is now private in Pri
        m_private = 3;      // not okay: derived classes can't access private members in the base class
    }
};

int main()
{
    // Outside access uses the access specifiers of the class being accessed.
    // In this case, the access specifiers of base.
    Base base;
    base.m_public = 1;      // okay: m_public is public in Base
    base.m_protected = 2;   // not okay: m_protected is protected in Base
    base.m_private = 3;     // not okay: m_private is private in Base

    Pri pri;
    pri.m_public = 1;       // not okay: m_public is now private in Pri
    pri.m_protected = 2;    // not okay: m_protected is now private in Pri
    pri.m_private = 3;      // not okay: m_private is inaccessible in Pri

    return 0;
}
```

86