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

Hazırlayan: M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasında İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Giriş

- **Polymorphism (çok biçimlilik),** bir sınıf nesnesinin başka bir sınıfın nesnesi gibi davranmasını sağlar.
- Bir sınıf nesnesi başka bir sınıfın veri üyelerine erişebilir.
- Bir sınıf nesnesi başka bir sınıfın fonksiyon üyelerine erişebilir.

3

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki

- **Kalıtım hiyerarşisindeki** tüm **nesnelerin sanki base class'ın bir nesnesi gibi davranmasını sağlar.**
- Bir program tanımlanan **base class pointer'ı ile derived class'tan oluşturulan nesnelerini gösterebilir.**

Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki

```
1  // Fig. 13.1: fig13_01.cpp
2  // Aiming base-class and derived-class pointers at base-class
3  // and derived-class objects, respectively.
4  #include <iostream>
5  #include <iomanip>
6  #include "CommissionEmployee.h"
7  #include "BasePlusCommissionEmployee.h"
8  using namespace std;
9
10 int main()
11
12     // create base-class object
13     CommissionEmployee commissionEmployee(
14         "Sue", "Jones", "222-22-2222", 10000, .06 );
15
16     // create base-class pointer
17     CommissionEmployee *commissionEmployeePtr = 0;
18
19     // create derived-class object
20     BasePlusCommissionEmployee basePlusCommissionEmployee(
21         "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
22
23     // create derived-class pointer
24     BasePlusCommissionEmployee *basePlusCommissionEmployeePtr = 0;
25
26     // set floating-point output formatting
27     cout << fixed << setprecision( 2 );
28
```

Temel Sınıflar ve Türetilmiş Sınıflar

```
29      // output objects commissionEmployee and basePlusCommissionEmployee
30      cout << "Print base-class and derived-class objects:\n\n";
31      commissionEmployee.print(); // invokes base-class print
32      cout << "\n\n";
33      basePlusCommissionEmployee.print(); // invokes derived-class print
34
35      // aim base-class pointer at base-class object and print
36      commissionEmployeePtr = &commissionEmployee; // perfectly natural
37      cout << "\n\n\nCalling print with base-class pointer to "
38           << "\nbase-class object invokes base-class print function:\n\n";
39      commissionEmployeePtr->print(); // invokes base-class print
40
41      // aim derived-class pointer at derived-class object and print
42      basePlusCommissionEmployeePtr = &basePlusCommissionEmployee; // natural
43      cout << "\n\n\nCalling print with derived-class pointer to "
44           << "\nderived-class object invokes derived-class "
45           << "print function:\n\n";
46      basePlusCommissionEmployeePtr->print(); // invokes derived-class print
47
48      // aim base-class pointer at derived-class object and print
49      commissionEmployeePtr = &basePlusCommissionEmployee;
50      cout << "\n\n\nCalling print with base-class pointer to "
51           << "derived-class object\ninvokes base-class print "
52           << "function on that derived-class object:\n\n";
53      commissionEmployeePtr->print(); // invokes base-class print
54      cout << endl;
55  } // end main
```

Base class pointer'ı ile türetilmiş class nesnesi gösteriliyor.

Temel Sınıflar ve Türetilmiş Sınıflar

```
Print base-class and derived-class objects:

commission employee: Sue Jones
social security number: 222-22-2222
gross sales: 10000.00
commission rate: 0.06

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 300.00


Calling print with base-class pointer to
base-class object invokes base-class print function:

commission employee: Sue Jones
social security number: 222-22-2222
gross sales: 10000.00
commission rate: 0.06


Calling print with derived-class pointer to
derived-class object invokes derived-class print function:

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 300.00


Calling print with base-class pointer to derived-class object
invokes base-class print function on that derived-class object:

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

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Derived Class Pointer'ı ve Base Class Nesnesi

- **Derived class pointer'ı ile base class nesnesi gösterilemez.**
- Türetilmiş sınıf pointer'ı kullanılarak temel sınıf nesnesindeki üyeler üzerinde işlem yapılamaz.

Derived Class Pointer'ı ve Base Class Nesnesi

```
1  // Fig. 13.2: fig13_02.cpp
2  // Aiming a derived-class pointer at a base-class object.
3  #include "CommissionEmployee.h"
4  #include "BasePlusCommissionEmployee.h"
5
6  int main()
7  {
8     CommissionEmployee commissionEmployee(
9        "Sue", "Jones", "222-22-2222", 10000, .06 );
10    BasePlusCommissionEmployee *basePlusCommissionEmployeePtr = 0;
11
12    // aim derived-class pointer at base-class object
13    // Error: a CommissionEmployee is not a BasePlusCommissionEmployee
14    basePlusCommissionEmployeePtr = &commissionEmployee;
15 } // end main
```

Microsoft Visual C++ compiler error message:

```
C:\cpphtp8_examples\ch13\Fig13_02\fig13_02.cpp(14) : error C2440: '=' :
   cannot convert from 'CommissionEmployee *' to 'BasePlusCommissionEmployee *'
        Cast from base to derived requires dynamic_cast or static_cast
```

11

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Base Class Pointer'ı ve Derived Class Fonksiyonu

- **Base class pointer'ı ile derived class'a ait fonksiyon çağrılamaz.**
- Derived class nesnesini gösteren base class pointer'ını derived class pointer'ına doğrudan dönüştürünce **(explicit casting),** sadece derived class'ta olan üyelere erişim yapılabilir **(downcasting).**
- Base class pointer'ı derived class nesnesini gösterebilir, ancak sadece base class fonksiyonlarını çağırabilir.

Base Class Pointer'ı ve Derived Class Fonksiyonu

- Base class pointer'ı ile derived class fonksiyonu çağrılamaz.

```
1  // Fig. 13.3: fig13_03  .cpp
2  // Attempting to invoke derived-class-only member functions
3  // through a base-class pointer.
4  #include "CommissionEmployee.h"
5  #include "BasePlusCommissionEmployee.h"
6
7  int main()
8  {
9     CommissionEmployee *commissionEmployeePtr = 0; // base class
10    BasePlusCommissionEmployee basePlusCommissionEmployee(
11       "Bob", "Lewis", "333-33-3333", 5000, .04, 300 ); // derived class
12
13    // aim base-class pointer at derived-class object
14    commissionEmployeePtr = &basePlusCommissionEmployee;
15
16    // invoke base-class member functions on derived-class
17    // object through base-class pointer (allowed)
18    string firstName = commissionEmployeePtr->getFirstName();
19    string lastName = commissionEmployeePtr->getLastName();
20    string ssn = commissionEmployeePtr->getSocialSecurityNumber();
21    double grossSales = commissionEmployeePtr->getGrossSales();
22    double commissionRate = commissionEmployeePtr->getCommissionRate();
23
24    // attempt to invoke derived-class-only member functions
25    // on derived-class object through base-class pointer (disallowed)
26    double baseSalary = commissionEmployeePtr->getBaseSalary();
27    commissionEmployeePtr->setBaseSalary( 500 );
28 } // end main
```

Base class pointer'ı ile türetilmiş class nesnesi gösteriliyor.

Base Class Pointer'ı ve Derived Class Fonksiyonu

- Base class pointer'ı ile derived class fonksiyonu çağrılamaz.

Microsoft Visual C++ compiler error messages:

```
C:\cpphtp8_examples\ch13\Fig13_03\fig13_03.cpp(26) : error C2039:
   'getBaseSalary' : is not a member of 'CommissionEmployee'
      C:\cpphtp8_examples\ch13\Fig13_03\CommissionEmployee.h(10) :
         see declaration of 'CommissionEmployee'
C:\cpphtp8_examples\ch13\Fig13_03\fig13_03.cpp(27) : error C2039:
   'setBaseSalary' : is not a member of 'CommissionEmployee'
      C:\cpphtp8_examples\ch13\Fig13_03\CommissionEmployee.h(10) :
         see declaration of 'CommissionEmployee'
```

GNU C++ compiler error messages:

```
fig13_03.cpp:26: error: 'getBaseSalary' undeclared (first use this function)
fig13_03.cpp:27: error: 'setBaseSalary' undeclared (first use this function)
```

15

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Upcasting ve Downcasting

Upcasting:

- Derived class nesnesinin base class pointer'ı ile gösterilmesidir.
- Güvenli olduğundan dolaylı (implicit) tür dönüşümü yapılır.

Downcasting:

- Derived class pointer'ı ile base class nesnesinin (derived class nesnesinin pointer'ı) gösterilmesidir.
- Güvenli olmadığından doğruda (explicit) tür dönüşümü yapılır.

17

Upcasting ve Downcasting

```
#include<iostream>
using namespace std;

class Base
{
public:
    int x;
    Base()
    {
        x = 5;
    }
    void show()
    {
        cout << "\n Base show function";
        cout << "\n x = " << x;
    }
};

class Derived : public Base
{
public :
    int y;
    Derived()
    {
        x = 15;
        y = 20;
    }

    void display()
    {
        cout << "\n Derived display function";
        cout << "\n x = " << x;
        cout << "\n y = " << y;
    }
};
```

18

Upcasting ve Downcasting

```
37  int main()
38  {
39      //Upcasting(Implicit)
40      Derived *dr = new Derived();
41      Base *base = dr;
42      base->show();
43      // base->display();//ERROR
44
45      //Donwcasting (Explicit)
46      Derived *derived = (Derived *)base;
47      derived->show();
48      derived->display();
49      return 0;
50  }
```

Türetilmiş class pointer'ı ile base class türündeki pointer (türetilmiş class nesnesini gösteriyor) gösteriliyor.

```
Base show function
x = 15
Base show function
x = 15
Derived display function
x = 15
y = 20
```

19

Konular

- Giriş
- Kalıtım Hiyerarşisinde Nesneler Arasındaki İlişki
- Derived Class Pointer'ı ve Base Class Nesnesi
- Base Class Pointer'ı ve Derived Class Fonksiyonu
- Upcasting ve Downcasting
- Virtual Fonksiyonlar

Virtual Fonksiyonlar

- Belirlenen tür, hangi sınıfın fonksiyonunun çalıştırılacağına karar verir.
- Önceki örnekte, `CommissionEmployee` pointer'ı üyesi olan `print()` fonksiyonu ile `BasePlusCommissionEmployee` nesnesinin değerlerini yazdırmaktadır.
- `BasePlusCommissionEmployee` pointer'ı üyesi olan `print()` fonksiyonu ile kendi değerlerini yazdırmaktadır.
- `virtual` fonksiyonlarla nesnenin türüne göre üye fonksiyon çalıştırılır.

Virtual Fonksiyonlar

```
1  // Fig. 13.4: 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
```

Virtual Fonksiyonlar

```
21     void setSocialSecurityNumber( const string & ); // set SSN
22     string getSocialSecurityNumber() const; // return SSN
23
24     void setGrossSales( double ); // set gross sales amount
25     double getGrossSales() const; // return gross sales amount
26
27     void setCommissionRate( double ); // set commission rate
28     double getCommissionRate() const; // return commission rate
29
30     virtual double earnings() const; // calculate earnings
31     virtual 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
```


Virtual Fonksiyonlar

```
1   // Fig. 13.5: 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     virtual double earnings() const; // calculate earnings
21     virtual void print() const; // print BasePlusCommissionEmployee object
22  private:
23     double baseSalary; // base salary
24  }; // end class BasePlusCommissionEmployee
25
26  #endif
```

Virtual Fonksiyonlar

```
1  // Fig. 13.6: fig13_06.cpp
2  // Introducing polymorphism, virtual functions and dynamic binding.
3  #include <iostream>
4  #include <iomanip>
5  #include "CommissionEmployee.h"
6  #include "BasePlusCommissionEmployee.h"
7  using namespace std;
8
9  int main()
10 {
11    // create base-class object
12    CommissionEmployee commissionEmployee(
13       "Sue", "Jones", "222-22-2222", 10000, .06 );
14
15    // create base-class pointer
16    CommissionEmployee *commissionEmployeePtr = 0;
17
18    // create derived-class object
19    BasePlusCommissionEmployee basePlusCommissionEmployee(
20       "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );
21
22    // create derived-class pointer
23    BasePlusCommissionEmployee *basePlusCommissionEmployeePtr = 0;
24
```


Virtual Fonksiyonlar

```
25    // set floating-point output formatting
26    cout << fixed << setprecision( 2 );
27
28    // output objects using static binding
29    cout << "Invoking print function on base-class and derived-class "
30       << "\nobjects with static binding\n\n";
31    commissionEmployee.print(); // static binding
32    cout << "\n\n";
33    basePlusCommissionEmployee.print(); // static binding
34
35    // output objects using dynamic binding
36    cout << "\n\n\nInvoking print function on base-class and "
37       << "derived-class \nobjects with dynamic binding";
38
39    // aim base-class pointer at base-class object and print
40    commissionEmployeePtr = &commissionEmployee;
41    cout << "\n\nCalling virtual function print with base-class pointer"
42       << "\nto base-class object invokes base-class "
43       << "print function:\n\n";
44    commissionEmployeePtr->print(); // invokes base-class print
45
```

Virtual Fonksiyonlar

```
46     // aim derived-class pointer at derived-class object and print
47     basePlusCommissionEmployeePtr = &basePlusCommissionEmployee;
48     cout << "\n\nCalling virtual function print with derived-class "
49        << "pointer\nto derived-class object invokes derived-class "
50        << "print function:\n\n";
51     basePlusCommissionEmployeePtr->print(); // invokes derived-class print
52
53     // aim base-class pointer at derived-class object and print
54     commissionEmployeePtr = &basePlusCommissionEmployee;
55     cout << "\n\nCalling virtual function print with base-class pointer"
56        << "\nto derived-class object invokes derived-class "
57        << "print function:\n\n";
58
59     // polymorphism; invokes BasePlusCommissionEmployee's print;
60     // base-class pointer to derived-class object
61     commissionEmployeePtr->print();
62     cout << endl;
63  } // end main
```


Virtual Fonksiyonlar

```
Invoking print function on base-class and derived-class
objects with static binding

commission employee: Sue Jones
social security number: 222-22-2222
gross sales: 10000.00
commission rate: 0.06

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 300.00
```

```
Invoking print function on base-class and derived-class
objects with dynamic binding

Calling virtual function print with base-class pointer
to base-class object invokes base-class print function:

commission employee: Sue Jones
social security number: 222-22-2222
gross sales: 10000.00
commission rate: 0.06
```

Virtual Fonksiyonlar

```
Calling virtual function print with derived-class pointer
to derived-class object invokes derived-class print function:

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 300.00
```

```
Calling virtual function print with base-class pointer
to derived-class object invokes derived-class print function:

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 300.00
```