

Basics: From C to C++

Computer Programming for Engineers (DSAF003-42)

Fall, 2021

Practice 7 : Inheritance

Instructor:

Youngjoong Ko (nlp.skku.edu)

Inheritance example

■ IS-A relationship (student is a person)

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  class Person{
5  private:
6      string name;
7      int age;
8  public:
9      Person(string myname,int myage):
10         name(myname), age(myage){}
11         void PrintPersonInfo(){
12             cout << "Hi, I'm " << name << " and "
13             << age << " years old." << endl;
14         }
15     };
16     class Student{
17     private:
18         string name;
19         int age;
20         string major;
21     public:
22         Student(string myname,int myage,string mymajor):
23             name(myname), age(myage), major(mymajor){}
24         void PrintStudentInfo(){
25             cout << "Hi, I'm " << name << " and "
26             << age << " years old." << endl;
27             cout << "And my major is " << major << endl;
28         }
29     };
30     int main(){
31         Person A("Minsu",22);
32         Student B("Taehun",26,"AI");
33
34         A.PrintPersonInfo();
35         cout << endl;
36         B.PrintStudentInfo();
37
38         return 0;
39     };
```



```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  class Person{
5  private:
6      string name;
7      int age;
8  public:
9      Person(string myname,int myage):
10         name(myname), age(myage){}
11         void PrintPersonInfo(){
12             cout << "Hi, I'm " << name << " and "
13             << age << " years old." << endl;
14         }
15     };
16     class Student : public Person{
17     private:
18         string major;
19     public:
20         Student(string myname,int myage,string mymajor):
21             Person(myname,myage), major(mymajor){}
22         void PrintStudentInfo(){
23             PrintPersonInfo();
24             cout << "And my major is " << major << endl;
25         }
26     };
27     int main(){
28         Person A("Minsu",22);
29         Student B("Taehun",26,"AI");
30
31         A.PrintPersonInfo();
32         cout << endl;
33         B.PrintStudentInfo();
34
35         return 0;
36     };
```

Hi, I'm Minsu and 22 years old.

Hi, I'm Taehun and 26 years old.
And my major is AI

Hi, I'm Minsu and 22 years old.

Hi, I'm Taehun and 26 years old.
And my major is AI

Private of base class

- Member function of derived class can not access private variables of base class directly

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  class Person{
5  private:
6      string name;
7      int age;
8  public:
9      Person(string myname,int myage):
10         name(myname), age(myage){}
11         void PrintInfo(){
12             cout << "Hi, I'm " << name << " and "
13             << age << " years old." << endl;
14         }
15         string getName() const {return name;}
16         int getage() const {return age;}
17         void changename(string newname){
18             name = newname;
19         }
20         void changeage(int newage){
21             age = newage;
22         }
23     };
```

```
24     class Student : public Person{
25     private:
26         string major;
27     public:
28         Student(string myname,int myage,string mymajor):
29             Person(myname,myage), major(mymajor){}
30         void PrintInfo(){
31             cout << "Hi, I'm " << getName() << "(" << getage()
32             << ")" << " and my major is " << major << endl;
33         }
34         string getmajor() const {return major;}
35         void changemajor(string newmajor){
36             major = newmajor;
37         }
38     };
39     int main(){
40         Student A("Taehun",26,"AI");
41
42         A.PrintInfo();
43         cout << endl;
44         A.changename("minsu");
45         A.changeage(20);
46         A.changemajor("software");
47         A.PrintInfo();
48
49         return 0;
50     };
```

Hi, I'm Taehun(26) and my major is AI

Hi, I'm minsu(20) and my major is software

Constructor of derived class

```
1  #include <iostream>
2  using namespace std;
3  class Base{
4  private: int baseNum;
5  public:
6      Base():baseNum(1){
7          cout<<"Base()"<<endl;
8      }
9      Base(int n):baseNum(n){
10         cout<<"Base(int n)"<<endl;
11     }
12     void printNum(){
13         cout<<"baseNum: "<<baseNum<<endl;
14     }
15 };
16 class Derived : public Base{
17 private: int derivedNum;
18 public:
19     Derived():derivedNum(10){
20         cout<<"DerivedBase()"<<endl;
21     }
22     Derived(int n):derivedNum(n){
23         cout<<"DerivedBase(int n)"<<endl;
24     }
25     Derived(int n1, int n2)
26     :Base(n1),derivedNum(n2){
27         cout<<"DerivedBase(int n1, int n2)"<<endl;
28     }
29     void printDerivedNum(){
30         printNum();
31         cout<<"derivedNum: "<<derivedNum<<endl;
32     }
33 };
```

```
34 int main(){
35     cout<<"---Case 1---"<<endl;
36     Derived dr1;
37     dr1.printDerivedNum();
38
39     cout<<"---Case 2---"<<endl;
40     Derived dr2(20);
41     dr2.printDerivedNum();
42
43     cout<<"---Case 3---"<<endl;
44     Derived dr3(2,30);
45     dr3.printDerivedNum();
46
47     return 0;
48 }
```

```
---Case 1---
Base()
DerivedBase()
baseNum: 1
derivedNum: 10
---Case 2---
Base()
DerivedBase(int n)
baseNum: 1
derivedNum: 20
---Case 3---
Base(int n)
DerivedBase(int n1, int n2)
baseNum: 2
derivedNum: 30
```

Exercise 1

■ Define the class BestStudent

- BestStudent is a derived class of Student

```
#include <iostream>
#include <string>
using namespace std;
class Person{
private:
    string name;
    int age;
public:
    Person(string myname,int myage):
        name(myname), age(myage){}
    void PrintInfo(){
        cout << "Hi, I'm " << name << " and "
        << age << " years old." << endl;
    }
    string getName() const {return name;}
    int getage() const {return age;}
    void changename(string newname){
        name = newname;
    }
    void changeage(int newage){
        age = newage;
    }
};
```

```
class Student : public Person{
private:
    string major;
public:
    Student(string myname,int myage,string mymajor):
        Person(myname,myage), major(mymajor){}
    void PrintInfo(){
        cout << "Hi, I'm " << getName() << "(" << getage()
        << ")" << " and my major is " << major << endl;
    }
    string getmajor() const {return major;}
    void changemajor(string newmajor){
        major = newmajor;
    }
};

int main(){
    BestStudent A("Taehun",26,"AI",95);

    A.PrintInfo();
    return 0;
};
```

Hi, I'm Taehun(26, AI) and my score is 95

Private, Protected, Public

- Private < protected < public (access level)
- Most use public inheritance

```
1  #include <iostream>
2  using namespace std;
3
4  class Base{
5  private:
6      int num1=1;
7  protected:
8      int num2=2;
9  public:
10     int num3=3;
11     void ShowData(){
12         cout << num1<< endl;
13         cout << num2<< endl;
14         cout << num3<< endl;
15     }
16 };
```

```
17 class public_Base : public Base{
18 public:
19     void ShowBaseMember(){
20         // cout << "num1: " << num1<< endl;
21         cout << "num2: " << num2<< endl;
22         cout << "num3: " << num3<< endl;
23     }
24 };
25 class protected_Base : protected Base{
26 public:
27     void ShowBaseMember(){
28         // cout << "num1: " << num1<< endl;
29         cout << "num2: " << num2<< endl;
30         cout << "num3: " << num3<< endl;
31     }
32 };
33 class private_Base : private Base{
34 public:
35     void ShowBaseMember(){
36         // cout << "num1: " << num1<< endl;
37         cout << "num2: " << num2<< endl;
38         cout << "num3: " << num3<< endl;
39     }
40 };
```

```
41 int main(){
42     public_Base A;
43     A.ShowBaseMember();
44     cout << "A's num3: " << A.num3 << endl;
45
46     protected_Base B;
47     B.ShowBaseMember();
48     // cout << "B's num3: " << B.num3 << endl;
49
50     private_Base C;
51     C.ShowBaseMember();
52     // cout << "C's num3: " << C.num3 << endl;
53
54     return 0;
55 }
```

```
num2: 2
num3: 3
A's num3: 3
num2: 2
num3: 3
num2: 2
num3: 3
```

Exercise 2

■ Define Player, MainPlayer, SubPlayer class

- MainPlayer and SubPlayer is a derived class of Player class and have showinfo() function
- MainPlayer's salary is inputted value.
- SubPlayer's salary is num of matches*pay per match (both are input)
- SubPlayer class has addmatch() function that increase num of matches

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

class Player{
private:
    string name;
};

class MainPlayer : public Player{
private:
    int salary;
};

class SubPlayer : public Player{
private:
    int matches;
    int pay_per_match;
};
```

```
int main(){
    MainPlayer a("태훈", 200);
    a.showinfo();

    SubPlayer b("충원",15,5);
    b.showinfo();
    b.addmatch(5);
    b.showinfo();

    return 0;
}
```

```
태훈's salary: 200
충원's salary: 75
충원's salary: 100
```

Assignment

■ Using Exercise2 define StarPlayer, PlayerList class

- StarPlayer is a derived class of MainPlayer
- StarPlayer have showinfo(), addmatch() functions
- StarPlayer's salary is inputted salary + num of matches*pay per match

■ PlayerList have addplayer(), showinfo(), showtotal() functions

- addplayer function add their player
- showinfo function call each player's showinfo()
- showtotal function print total salary of all players

```
int main(){
    PlayerList players;
    players.addplayer(new MainPlayer("태훈",200));
    players.addplayer(new MainPlayer("준희",300));

    SubPlayer* b = new SubPlayer("충원",15,5);
    b->addmatch(5);
    players.addplayer(b);

    StarPlayer* c = new StarPlayer("두영",500,30,10);
    c->addmatch(10);
    players.addplayer(c);

    players.showinfo();
    players.showtotal();

    return 0;
}
```

```
태훈's salary: 200
준희's salary: 300
충원's salary: 100
두영's salary: 900
Total Salary: 1500
```