UEE1303(1070) S12: Object-Oriented Programming

Inheritance (I)

What you will learn from Lab 9

In this laboratory, you will learn the concept of inheritance and its usage.

TASK 9-1 EXAMPLE OF INHERITANCE

✓ Please compiler and execute the program `lab9-1`, where `Point4D` is a driven class from the base class `Point2D`.

```
// lab9-1-1.cpp
#include <iostream>
using std::cout;   using std::endl;

class Point2D
{
private:
    int x;
    int y;

public:
    Point2D(int n1 = 0, int n2 = 0):x(n1), y(n2){}
    void display() const;
};
class Point4D : public Point2D
{
private:
    int z;
    int t;
public:
    Point4D(int n1 =0,int n2 = 0,int n3 = 0,int n4 = 0):Point2D(n1,n2),z(n3), t(n4){}
    void display() const;
};
void Point4D::display() const
{
    Point2D::display();
    cout << "," << z << "," << t;
}

int main()
{
    Point4D pt(1,2,3,4);
    pt.display(); cout << endl;

    return 0;
}
```

- Note that `Point4D` has member of class `Point2D` in addition to its own members.
- You can put the constructor of the base class in the initialization list for the driven class.
- Note that member function of a derived class cannot access the private part of a base class. For example, the function `Point4D::display()` cannot be defined as

```
void Point4D::display() const
{
   cout << x << "," << y;       // x and y are inaccessible
   cout << "," << z << "," << t;
}
```

The hidden member `x` and `y` of the derived class `Point4D` is accessible through the public member function `Point2D::display();`.

You can define *accessor* and *mutator* functions in `Point2D` to access private members.

✓ Please compiler and execute the program `lab9-1-2`

```
// lab9-1-2.cpp

/*  The Point2D and Point4D class defined in lab9-1-1  */

int main()
{
   Point2D pt2(3,4);

   Point4D pt4(1,2,3,4);
   pt4.display(); cout << endl;

   pt2 = pt4;  // OK, every Point2D is a Point4D
   pt2.display(); cout << endl;

   pt4 = pt2;  // Error, not every Ponint4D is a Point2D
   pt4.display(); cout << endl;

   return 0;
}
```

- You can comment the incorrect lines to observe the results.
- If you require type conversion from a base class to driven class (eg. `pt4 = pt2`), you have to provide additional member functions of `Point4D` to achieve it.

✓ Please compiler and execute the program `lab9-1-3`

```
// lab9-1-3.cpp

/*  The Point2D and Point4D class defined in lab9-1-1  */
```

```
void f(const Point2D &p1, const Point2D &p2)
{
   p1.display(); cout << endl;
   p2.display(); cout << endl;
}

int main()
{
   Point2D pt2(3,4);
   Point4D pt4(1,2,3,4);

   f(pt2,pt4);

   return 0;
}
```

- Note that the prototype of function `f` is `void f(const Point2D &, const Point2D &)`.

TASK 9-2 CLASS HIERARCHY

- ✓ A derived class can be a base class of another derived class.

```
// lab10-3.cpp

/*  The Point2D and Point4D class defined in lab9-1-1  */

class Car : public Point4D
{
private:
   int color;
   int year;
public:
   Car(int n1 =0, int n2 = 0, int n3 = 0, int n4 = 0):Point4D(n1,n2,n3,n4)
   {
      color = 0;
      year = 0;
   }
   Car(const Point4D &p):Point4D(p){color = 0; year = 0;}  // copy constructor

   void display() const;
   void setColor(const int c){color = c;}
   void setYear(const int y){year = y;}
};

void Car::display() const
{
   cout << "color: " << color << endl;
   cout << "year: " << year << endl;
   Point4D::display();
```

```
}

int main()
{
   Point4D pt4(1,2,3,4);

   Car c1(pt4);
   c1.setColor(128);
   c1.setYear(2011);
   c1.display(); cout << endl;

   return 0;
}
```

- Note that, to enable copy constructor of `Car`, you should also provide copy constructor for `Point2D` and `Point4D`.

TASK 9-3 EXERCISE

1. *LAB 9-1

✓ Please modify the class `Point2D` and `Point4D` defined in `lab9-1`. In `Point2D`, the member `x` and `y` become two pointers to integer, respectively. Similarly, the member `z` and `t` should be changed as pointers. The modified classes are shown as follows,

```
// Point2D.h
#ifndef POINT2D_H
#define POINT2D_H

class Point2D
{
private:
   int *x;
   int *y;

public:

};

#endif
```

✓ Please implement `Point2D` and `Point4D` in different files

```
// Point4D.h
#ifndef POINT4D_H
#define POINT4D_H

#include "Point2D.h"
```

```
class Point4D
{
private:
    int *z;
    int *t;

public:

};
#endif
```

✓ Please finish the remaining part to make the following main function work successfully.

```
#include <iostream>
#include "Point2D.h"
#include "Point4D.h"
using std::cout;   using std::endl;

int main()
{
    Point2D pt1(1,2);
    Point2D pt2(3,4);

    pt1.display(); cout << endl;
    pt2.display(); cout << endl;

    pt2 = pt1;
    pt2.display(); cout << endl;

    Point4D pt4(5,6,7,8);
    pt4.display(); cout << endl;

    pt2 = pt4;
    pt2.display(); cout << endl;

    pt4 = pt1;
    pt4.display(); cout << endl; //pt4 could be (1,2,7,8) or (1,2,0,0)
    return 0;
}
```

2. PACKAGE-DELIVERY SERVICES

✓ Package-delivery services offer a number of different shipping options, each with specific costs associated. Create an **inheritance hierarchy** to represent **various types of packages**. Use `Package` as the base class of the hierarchy, and then include classes `TwoDayPackage` and `OvernightPackage` that derive from `Package`. Base class `Package` should include data members representing the `name` and `city` for both the sender and the recipient of the package, in addition to data members that store the `weight` and `costPerWeight` to ship the package. `Package`'s constructor should initialize these data members. Ensure that the `weight` and

costPerWeight contain positive values. Package should provide a public member function calculateCost that returns a double indicating the cost associated with shipping the package. Package's calculateCost function should determine the cost by multiplying the weight by the cost per weight. Derived class TwoDayPackage should inherit the functionality of base class Package, but also include a data member that represents a flat fee that the shipping company charges for two-day-delivery service. TwoDayPackage's constructor should receive a value to initialize this data member. TwoDayPackage should redefine member function calculateCost so that it computes the shipping cost by adding the flat fee to the cost calculated by class Package's calculateCost function. Class OvernightPackage should inherit directly from class Package and contain an additional data member representing an additional fee per weight charged for overnight-delivery service. OvernightPackage should redefine member function calculateCost so that it adds the additional fee per weight costOvernightPerWeight to the standard costPerWeight before calculating the shipping cost.

✓ Sample output:

```
Package 1:
Sender:
Lou Brown / Boston
Recipient:
Mary Smith / New York
Cost: $4.25

Package 2:
Sender:
Lisa Klein / Somerville
Recipient:
Bob George / Cambridge
Cost: $8.825

Package 3:
Sender:
Ed Lewis / Boston
Recipient:
Don Kelly / Denver
Cost: $11.6375
```

✓ Please finish the program "Package.h".

```
// Package.h
```

```
#ifndef PACKAGE_H
#define PACKAGE_H

#include <string>
using std::string;
class Package
{
private:

   string senderName;
   string senderCity;
   string recipientName;
   string recipientCity;

   double weight;        // weight of the package
   double costPerWeight; // cost per weight to ship the package

public:
   /* any member functions if necessary */
};

class TwoDayPackage : public Package
{
private:
    double flatFee; // flat fee for two-day-delivery service

public:
   /* any member functions if necessary */
};

class OvernightPackage : public Package
{
private:
    double overnightFreePerWeight; // flat fee weight for overnight delivery

public:
   /* any member functions if necessary */
};

#endif
```

✓ Your main program should be like as,

```
#include <iostream>
using std::cout;  using std::endl;
#include "Package.h" // Package class definition
int main()
{
    Package package1("Lou Brown", "Boston","Mary Smith", "New York", 8.5, .5 );
    TwoDayPackage package2("Lisa Klein", "Somerville",
                           "Bob George", "Cambridge", 10.5, .65, 2.0 );
```

```
    OvernightPackage package3("E Lewis", "Boston", "Don Kelly", "Denver",
                              12.25, .7, .25 );

    /* display the package's information */

    return 0;
}
```