Bài thực hành số 3

Mục tiêu:

- Đơn kế thừa (single inheritance)
- Đa kế thừa (multiple inheritance)

1. Đơn kế thừa (kế thừa đơn)

Chúng ta sử dụng kế thừa trong lập trình hướng đối tượng để có thể sử dụng lại các chức năng của các lớp khác. Các thành phần của lớp dẫn xuất chỉ có thể truy xuất đến các thành phần của lớp cơ sở có khai báo là public hoặc protected.

- Tạo file mới
- Gõ đoạn chương trình sau:

```
#include <iostream>
using namespace std;
class vehicle
{
      protected:
            int wheels;
            float weight;
      public:
            void init(int whls, float wght);
            int get_wheels() {return wheels; }
            float get_weight() { return weight; }
            float wheel_loading() {return weight/wheels; }
};
class car: public vehicle {
            int passenger_load;
      public:
            void init(int whls,float wght,int people = 4);
            int passengers() { return  passenger_load; }
};
class truck: public vehicle {
            int passenger_load;
            float payload;
      public:
            void init_truck(int number= 2, float max_load= 24000.0);
            float efficiency();
            int passengers() { return passenger_load; }
};
void vehicle::init(int whls, float wght) {
      wheels = whls;
      weight = wght;
}
void car:: init(int whls, float wght, int people) {
      passenger_load = people;
      wheels = whls;
      weight = wght;
}
```

```
void truck:: init_truck(int number, float max_load) {
      passenger_load = number;
      payload = max_load;
}
float truck::efficiency(){
      return payload/(payload + weight);
}

int main(){
      vehicle bi;
      bi.init(2,25);
      cout << "Xe dap co"<<bi.get_wheels() << "banh xe. \n";
      cout << "Khoi dong banh xe "<< bicycle.wheel_loading()
             << "don vi khoi luong tren mot banh xe. \n";
      cout<<"K/luong xe la "<<bi.get_weight()<<"don vi khoi luong.";
      car c;
      c.init(4,1500.0,5);
      cout<<"Xe o to cho duoc "<<c.passengers()<<"hanh khach.\n";
      cout<<"Xe o to nang "<<c.get_weight()<<"don vi khoi luong.\n";
      cout<<"Khoi dong banh xe "<<c.wheel_loading()
                          << "don vi khoi luong tren mot banh xe.";
      truck v;
      v.init(18,7000.0);
      v.init_truck(1,15000.0);
      cout<<"Xe tai nang "<<v.get_weight();
      cout<<" don vi khoi  luong.\n";
      cout<<"Su hieu qua: "<< 100.0*v.efficiency()<<" phan tram.\n";
}
```

- Lưu file với tên "single51.cpp"
- Dịch và chạy chương trình

Kết quả:

```
Xe dap co 2 banh xe.
Khoi dong banh xe 12.5 don vi khoi luong tren mot banh xe.
K/luong cua xe la 25 don vi khoi luong.
Chiec xe o to cho duoc 5 hanh khach.
Chiec xe o to nang 1500 don vi khoi luong.
Khoi dong banh xe 375 don vi khoi luong tren mot banh xe.
Xe tai nang 7000 don vi khoi luong.
Su hieu qua:  68.181818 phan tram
```

Trong chương trình trên lớp vehicle có hai thành phần dữ liệu khai báo là **protected**, và nhiều hàm có khai báo là public. Lớp vehicle là lớp cơ sở của hai lớp car và truck. Chúng ta thấy trong hai lớp này có truy xuất đến lớp cơ sở.

2. Đa kế thừa

Đa kế thừa được sử dụng khi một lớp muốn kế thừa từ nhiều lớp cơ sở.

- Tạo file mới
- Gõ đoạn chương trình sau:

```
#include <iostream>
#include<iomanip.h>
using namespace std;
class person                        //first base class
{
            char sex, name[25];
            int roll_no;
      public:
            void getinfo();
            void display();
};

class academics                     //second base class
{
      private:
            char course_name[25];
            int semester;
            char grade[3];
      public:
            void getinfo();
            void display();
};

class stud_scholarship: public person, public academics
{
      private:
            float amount;
      public:
            void getinfo();
            void display();
};
void person::getinfo() {
      cout << "Ten: ";  cin>> name;
      cout<<"So: ";    cin>>roll_no;
      cout<<"Gioi tinh(F/M) : "; cin>> sex;
}
void person:: display()
{
      cout<<name<<"\t";
      cout<<roll_no<<"\t";
      cout<<sex<<"\t";
}

void academics::getinfo()
{
      cout<<"Ten khoa (BA/MBA/MCA etc)? "; cin>>course_name;
      cout<< "Hoc ky (1/2/3/...)? ";       cin>>semester;
      cout<<"muc do (A,B,B+,B-..) ? ";     cin>>grade;
}
void academics::display(){
      cout<<course_name<<"\t";
      cout<<semester<<"\t";
      cout<<grade<<"\t";
}
```

```
void stud_scholarship::getinfo()
{
      person::getinfo();     // thong tin ve person
      academics::getinfo(); // thong tin ve academics
      cout<<"Su ho tro ";
      cin>>amount;           // moi bo sung trong stud_scholarship
}

void stud_scholarship::display()
{
      person::display();
      academics::display();
      cout<<amount<<endl;
}

int main()
{
      stud_scholarship obj;
      cout<<"Nhap cac thong tin sau: "<<endl;
      obj.getinfo();
      cout<<endl;
      cout<<"Ten   So  G/tinh   Khoa    Hoc ky    Muc do   S/luong";
      cout<<endl;
      obj.display();
}
```

- Lưu file với tên "multi52.cpp"
- Dịch và chạy chương trình

Trong chương trình trên, các lớp cơ sở và lớp dẫn xuất có các hàm trùng tên vì vậy khi lớp dẫn xuất gọi các hàm này phải chỉ rõ là hàm được gọi của lớp nào (xem các hàm getinfo() và display() của lớp dẫn xuất).

Kết quả:

```
Nhap cac thong tin sau:
Ten: Cuong
So: 12
Gioi tinh(F/M) : M
Ten khoa (BA/MBA/MCA etc)? BA
Hoc ky (1/2/3/...)? 2
muc do (A,B,B+,B-..) ? A
Su ho tro 1000

Ten     So  G/tinh   Khoa    Hoc ky    Muc do  S/luong
Cuong   12      M       BA       2        A       1000

--------------------------------
```

3. Hàm khởi tạo trong kế thừa

Khi trong lớp cơ sở và lớp dẫn xuất có hàm khởi tạo (constructors) thì lớp cơ sở sẽ được khởi tạo trước với hàm khởi tạo mặc định hoặc hàm có tham số tùy thuộc vào khai báo ở trong lớp dẫn xuất.

- Tạo file mới
- Gõ đoạn chương trình sau:

```
#include<iostream>
class A{
public:
      A()   { cout<< "A"; }
       ~A()  { cout<< "~A"; }
};
class B:public A{
public:
      B()   { cout<< "B"; }
       ~B()  { cout<< "~B"; }
};


class C: public A {
public:
   C()      { cout<< "C"; }
     ~C()  { cout<< "~C" }
};
class D: public C, public B {
      B b;
public:
      D()  { cout<< "D" ; }
  ~D()  { cout<< "~D" }
};

int main()
{
      A a;  cout<<endl;
      B b;  cout<<endl;
      C c;  cout<<endl;
      D d;  cout<<endl;
}
```

- Lưu file với tên "multi53.cpp"
- Dịch và chạy chương trình

Kết quả:

```
A
A B
A C
A C A B A B D
~D ~B ~A ~B ~A ~C ~A ~C ~A ~B ~A ~A
```

Bài tập làm thêm

TH3.1.

1. Định nghĩa cấu trúc (struct) Date gồm các thuộc tính: day, month, year.
2. Định nghĩa lớp (class) Person gồm các thuộc tính: string name, Date dob;
3. Định nghĩa lớp Student kế thừa lớp Person và được bổ sung các thuộc tính: string address, email, telephone.
4. Định nghĩa lớp Undergrad kế thừa lớp Student và được bổ sung các thuộc tính: string school, major; person parents.
5. Định nghĩa lớp Degree gồm các thuộc tính: string university, discipline; Date awarded.
6. Định nghĩa lớp GradStudent kế thừa lớp Student và bổ sung các thuộc tính: string dept; Degree lastDegree.

Viết chương trình thực hiện các công việc sau:

- Nhập một danh sách sinh viên đại học (Undergrad), in các thông tin của những sinh viên đó ra màn hình.
- In danh sách các sinh viên học ngành (major) là CNTT.
- In danh sách các sinh viên thuộc một trường (school) nào đó (tên trường nhập từ bàn phím).
- Tìm thông tin của một sinh viên (tên sinh viên nhập từ bàn phím).
- Nhập một danh sách học viên cao học (GradStudent), in các thông tin của những học viên đó ra màn hình.
- In danh sách các học viên có bằng cấp trước đây (Degree) ngành (discipline) là CNTT và được cấp sau năm 2010.
- In danh sách các học viên của khoa (department) CNTT.

TH3.2. Hãy định nghĩa một lớp Shape (Hình) trong đó dữ liệu là 2 cạnh (width, và height). Dẫn xuất từ lớp Hình hai lớp con là lớp Triangle (Hình tam giác) và lớp Rectangle (Hình chữ nhật), trong mỗi lớp có phương thức tính diện tích area(). Trong hàm main() khai báo hai biến đối tượng của lớp Triangle và lớp Rectangle, sau đó gọi hàm area() để tính diện tích mỗi hình.

Yêu cầu: Viết hai phiên bản cho hai trường hợp:
1) Dữ liệu của Shape là protected, dùng hàm thành phần để nhập dữ liệu.
2) Dữ liệu của Shape là private, dùng hàm tạo có đối để khởi tạo dữ liệu.

TH3.3. Xây dựng lớp NGUOI, gồm các thuộc tính (private): họ tên, tuổi; các phương thức (public): nhap( ), xuat( ) và các phương thức khác.
Xây dựng lớp QLNV dẫn xuất từ lớp NGUOI, gồm các thuộc tính (private): snct (số năm công tác) và hs (hệ số lương); các phương thức (public): tính tiền lương, biết tienluong = lcb*hs + phucap, trong đó: lcb (lương cơ bản) là 1,5 triệu cho tất cả nhân viên, phucap = 0,2 triệu * snct.
Viết hàm main( ) trong đó:

- Nhập n nhân viên
- Tìm nhân viên có số năm công tác nhiều nhất
- In danh sách nhân viên theo thứ tự lương từ cao đến thấp

TH3.4. Xây dựng lớp SP (Số Phức), gồm các thuộc tính (private): phần thực và phần ảo; các phương thức (public): hàm tạo không đối và hàm tạo có hai đối để tạo và khởi gán số phức, các hàm nhập và xuất một số phức và định nghĩa chồng phép toán ! để tính tổng bình phương của phần thực và phần ảo (ví dụ: nếu z = 3 – 4i thì giá trị !z bằng 25).
Viết tiếp hàm main( ) trong đó:
- Nhập hai số phức s_1 và s_2, trong đó nhập s_1 dùng hàm tạo có đối, nhập s_2 dùng hàm nhập.
- Tính tổng s = $!s_1 + !s_2$.
- In các số phức s_1, s_2 và tổng s ra màn hình (mỗi số in trên một dòng).

TH3.5. Xây dựng lớp Nguoi (Người) gồm các thuộc tính gồm họ tên, tuổi và các phương thức cần thiết. Xây dựng lớp Quanly (Quản lý) dẫn xuất từ lớp Nguoi, bổ sung thêm thuộc tính snct (số năm công tác), hsl (hệ số lương). Công thức tính lương là: tienluong = lcb*hsl + phucap, trong đó: phucap = 3 triệu nếu snct >= 15 năm, phucap = 2 triệu nếu 5 =< snct < 15 năm và phucap = 1 triệu nếu snct<5 năm.

Viết hàm main( ) thực hiện:
- Nhập vào n nhân viên.
- Tìm nhân viên có tiền lương cao nhất và nhân viên có tiền lương thấp nhất.

TH3.6. Xây dựng lớp Xe, trong đó có các thuộc tính gồm: Biển số xe và trọng lượng của xe (tính theo tấn). Xây dựng lớp XeCon dẫn xuất từ lớp Xe, trong đó định nghĩa thêm thuộc tính: Số chỗ ngồi. Trong lớp trên có thể định nghĩa thêm một số hàm thành phần nếu cần thiết.

Viết chương trình:
- Nhập thông tin cho n xe con và m xe tải (n và m được nhập từ bàn phím).
- In danh sách các xe con có trọng lượng dưới 1 tấn và có không quá 5 chỗ ngồi.

TH3.7. Xây dựng lớp MH (Mặt Hàng), trong đó có các thuộc tính gồm: Mã hàng, tên hàng, nhà sản xuất, số lượng, đơn giá và một số phương thức cần thiết. Xây dựng lớp MayTinh (Máy Tính) dẫn xuất từ lớp MH, trong đó định nghĩa thêm các thuộc tính: Loại CPU, hệ điều hành, trọng lượng và một số hàm nếu cần.

Viết chương trình:
- Nhập thông tin cho n đối tượng của lớp MayTinh.
- In danh sách các máy tính có đơn giá cao nhất.