# BÀI THỰC TẬP 05 – LẬP TRÌNH CÀI ĐẶT ĐA KẾ THỪA

## A. MỤC TIÊU

- Hiểu khái niệm đa kế thừa, cú pháp cài đặt đa kế thừa.

- Biết cách xây dựng các lớp sử dụng đa kế thừa.

- Vận dụng lớp cơ sở ảo trong xây dựng bài toán đa kế thừa.

## B. NỘI DUNG

* Khai báo định nghĩa lớp đa kế thừa.
* Định nghĩa, sử dụng lớp cơ sở ảo xây dựng các chương trình.

## C. YÊU CẦU PHẦN CỨNG, PHẦN MỀM

* Phần cứng: Dung lượng RAM từ 1GB
* Phần mềm: Hệ điều hành Windows, C-free 5.0

## D. KẾT QUẢ SAU KHI HOÀN THÀNH

Xây dựng được ứng dụng với nhiều lớp được tổ chức theo sự phân cấp đa kế thừa.

## E. TÓM TẮT LÝ THUYẾT

1. **Đa kế thừa**

Cú pháp khai báo đa kế thừa như sau:

|  |
| --- |
| **class** <*Tên\_lớp\_dẫn\_xuất*> **:** <*Kiểu \_kế thừa*> <*tên\_lớp\_cơ\_sở\_1*> **,** <*Kiểu \_kế thừa*> <*tên\_lớp\_cơ\_sở\_2*>, …  **{**  ……………… **};** |

* Trong đa kế thừa, khi liệt kê các lớp cơ sở cần kế thừa, mỗi lớp được phân cách nhau bởi dấu phẩy “,”.
* Mỗi lớp cơ sở có thể có một kiểu dẫn xuất bởi một từ khoá dẫn xuất khác nhau.
* Nguyên tắc truy nhập vào các thành phần lớp cơ sở cũng hoàn toàn tương tự như trong kế thừa đơn
* *Thứ tự gọi các hàm tạo:* Các hàm tạo của các lớp cơ sở theo thứ tự khai báo của các lớp cơ sở trong lớp kế thừa lần lượt được gọi, sau cùng là hàm tạo của lớp kế thừa mới được gọi.
* *Thứ tự gọi các hàm hủy:* Các hàm hủy được gọi theo thứ tự ngược lại với cách gọi hàm tạo.
* *Truy nhập thành phần lớp cơ sở:* Giống như trong đơn kế thừa, trong hàm thành phần của lớp dẫn xuất có thể sử dụng tất cả các hàm thành phần **public** (hoặc **protected**) của lớp cơ sở.
* *Cách gọi hàm thành phần của lớp cơ sở:*
* Việc truy nhập từ đối tượng lớp dẫn xuất đến các thành phần của mỗi lớp cơ sở được tuân theo quy tắc phạm vi tương tự như trong đơn kế thừa.
* Trong trường hợp các lớp cơ sở đều có các thành phần cùng tên, việc truy xuất đến thành phần của lớp nào phải được chỉ rõ bằng toán tử phạm vi: “<Tên lớp>::” đối với thành phần lớp cơ sở đó.

1. **Lớp cơ sở ảo**

Trong đa kế thừa, thường gặp tình huống một lớp cơ sở có nhiều lớp kế thừa trực tiếp như hình sau:

**A**

**B**

**C**

**D**

Hình 5. 1 Sơ đồ lớp đa kế thừa sử dụng lớp cơ sở ảo

|  |
| --- |
| class A{  ………  int v1;  ……………  }  class B: public A{ }  class C: public A{ }  class D: public B, public C  {  } |

Lớp D kế thừa lớp A hai lần, như vậy các thành phần (hàm, dữ liệu) sẽ xuất hiện trong D hai lần. Các thành phần dữ liệu bị lặp lại trong các đối tượng (thành phần dữ liệu của mỗi đối tượng là độc lập). Để khắc phục điều này, khai báo lớp cơ sở A là lớp cơ sở ảo của lớp B, C.

Trong khai báo lớp dẫn xuất B,C bổ sung trước tên lớp kế thừa từ khóa **virtual.**

|  |
| --- |
| class A{  ………  int v1;  ……………  }  class B: public virtual A{ }  class C: public virtual A{ }  class D: public B, public C { } |

## F. HƯỚNG DẪN CHI TIẾT

**Bài 1:** Xây dựng lớp môn học gồm:

- Thuộc tính: tên môn học, số tiết

- Phương thức: Tạo không tham số, tạo có tham số, hủy, hiển thị thông tin một môn học.

Xây dựng lớp người gồm:

- Thuộc tính: họ tên, năm sinh

- Phương thức: Tạo không tham số, tạo có tham số, hủy, hiển thị thông tin một người.

Xây dựng giáo viên kế thừa từ lớp hai lớp môn học và người và bổ sung thêm:

- Thuộc tính: bộ môn

- Phương thức: Tạo không tham số, tạo có tham số, hủy, hiển thị thông tin một giáo viên.

Viết chương trình tạo một giáo viên hiên thị thông tin giáo viên đó.[2]

**Hướng dẫn:** Bài thực hành cung cấp cho sinh viên kiến thức về cài đặt đa kế thừa. Cách khai báo cài đặt lớp dẫn xuất từ nhiều lớp, cách viết hàm tạo trong lớp dẫn xuất.

Mô tả bài toán bằng sơ đồ kế thừa sau:

|  |
| --- |
| **MON\_HOC** |
| - monhoc: char\*  - st: int |
| + MON\_HOC();  + MON\_HOC(char, int);  + ~ MON\_HOC();  + void in(); |

|  |
| --- |
| **NGUOI** |
| - ht: char[]  - ns: int |
| + Nguoi()  + Nguoi(char, int)  + ~Nguoi()  + void in() |

|  |
| --- |
| **GIAO\_VIEN** |
| - bomon: char[] |
| + GIAO\_VIEN()  + GIAO\_VIEN(char, int, char, int, char)  + ~ GIAO\_VIEN()  + void in(); |

**Bước 1**: Viết lớp MON\_HOC mô tả thông tin môn học

|  |
| --- |
| **#include <iostream.h>**  **#include <string.h>**  **class MON\_HOC**  {  char \*monhoc;  int st;  **public:**  **MON\_HOC()**  {  monhoc=NULL; st=0;  }  **MON\_HOC(char \*monhoc1, int st1)**  {  int n = strlen(monhoc1);  monhoc = new char[n+1];  strcpy(monhoc,monhoc1);  st=st1;  }  **~ MON\_HOC()**  {  if (monhoc!=NULL)  {  delete monhoc;  st= 0;  }  }  **void in()**  {  cout << "\nTen mon: " << monhoc << " so tiet: " << st;  }  } ; |

**Bước 2:** Tương tự viết lớp Nguoi mô tả thông tin người.

|  |
| --- |
| **class NGUOI**  {  char \*ht;  int ns;  **public:**  **NGUOI();**  **NGUOI(char \*ht1, int ns1)**  {  int n = strlen(ht1);  ht = new char[n+1];  strcpy(ht,ht1);  ns=ns1;  }  **~NGUOI();**  **void in();**  }; |

**Bước 3:** Viết lớp GIAO\_VIEN kế thừa từ hai lớp trên.

|  |
| --- |
| **class GIAO\_VIEN : public NGUOI, public MON\_HOC**  {  char \*bomon;  **public:**  **GIAO\_VIEN():MON\_HOC(),NGUOI()**//Su dung ham tao khong doi  {  bomon=NULL;  }  // Hàm tạo có tham số, gọi lại hàm tạo hai lớp cơ sở.  ***GIAO\_VIEN(char \*ht1, int ns1, char \*monhoc1, int st1, char \*bomon1 ): NGUOI(ht1,ns1),MON\_HOC(monhoc1, st1)***  {  int n = strlen(bomon1);  bomon = new char[n+1];  strcpy(bomon,bomon1);  }  **~GIAO\_VIEN()**  {  if (bomon!=NULL)  delete bomon;  }  **void in()**  {  NGUOI::in();  cout << "\n Cong tac tai bo mon: " << bomon;  MON\_HOC::in();  }  }; |

**Bước 4:** Viết hàm main() thực hiện yêu cầu đề bài.

|  |
| --- |
| **main()**  {  GIAO\_VIEN \*g2;  //Goi toi cac ham tao co doi  g2 = new GIAO\_VIEN("Nguyen Van A", 1982, "CNTT", 40, "HTTT");  g2->in();  delete g2;  } |

**Bài 2:** Xây dựng lớp Người gồm các thuộc tính: Họ tên, năm sinh. Lớp bổ sung thêm các hàm thành phần:

- Hàm tạo, hàm hủy.

- Định nghĩa toán tử nhập thông tin một người.

- Định nghĩa toán tử xuất thông tin một người.

Xây dựng lớp Sinh viên (SV) gồm các thuộc tính: hạnh kiểm, điểm. Lớp này bổ sung thêm các hàm thành phần:

- Hàm tạo, hàm hủy

- Định nghĩa toán tử nhâp >>

- Định nghĩa toán tử xuất <<

Xây dựng lớp Sinh viên tại chức (SVTC) kế thừa từ hai lớp Người và lớp Sinh viên và bổ sung thêm các thuộc tính: học phí. Lớp bổ sung thêm các hàm thành phần:

- Định nghĩa toán tử nhâp >>

- Định nghĩa toán tử xuất <<

- Hàm tính học bổng theo điều kiện: Năm triệu nếu sinh viên đạt hạnh kiểm tốt và điểm từ 8.0. Ba triệu nếu sinh viên đạt hạnh kiểm khá trở lên và điểm từ 7.0. Hai triệu nếu sinh viên đạt hạnh kiểm khá trở lên và điểm từ 7.0. Còn lại học bổng bằng không.

Viết chương trình nhập vào danh sách n sinh viên tại chức (n nhập từ bàn phím), hiển thị thông tin đã nhập. Tìm sinh viên đạt học bổng. Sắp xếp danh sách sinh viên tại chức theo chiều tăng dần của điểm.

**Hướng dẫn:**

Bài thực hành cung cấp cho sinh viên kiến thức về cài đặt đa kế thừa. Cách khai báo cài đặt lớp dẫn xuất từ nhiều lớp, cách viết hàm tạo trong lớp dẫn xuất, định nghĩa toán tử nhập xuất trong lớp dẫn xuất.

Xây dựng sơ đồ lớp kế thừa như sau:

|  |
| --- |
| **SV** |
| - hk: char[]  - dtb: float |
| + SV()  + SV(char, float);  +istream & operator >>()  + xuat(): void |

|  |
| --- |
| **NGUOI** |
| - ht: char[]  - ns: int |
| + Nguoi()  + Nguoi(char, int)  + istream & operator >>()  + void xuat() |

|  |
| --- |
| **SVTC** |
| - hp: float |
| + SVTC()  + SVTC(char, int,char, float, float)  + hocbong()  + istream & operator >>()  + ostream & operator <<() |

**Bước 1:** Viết lớp Nguoi mô tả thông tin người.

|  |
| --- |
| **#include <iostream.h>**  **#include <string.h>**  **class Nguoi**  {  **private:**  char hoten[30];  int ns;  **public:**  **Nguoi()**  {  strcpy(hoten,"");  ns=0;  }  **Nguoi( char h[],int a)**  {  strcpy(hoten,h);  ns=a;  }  **friend istream & operator >>(istream &is, Nguoi &p)**  {  is.ignore(1);  cout<<"\n Nhap Ho ten: "; is.getline(p.hoten,30);  cout<<"\n Nhap nam sinh: "; is>>p.ns;  return is;  }  **void Xuat()**  {  cout<<"\n Ho ten: "<<hoten;  cout<<"\n Nam sinh: "<<ns;  }  }; |

**Bước 2:** Viết lớp SV mô tả thông tin của các đối tượng sinh viên

|  |
| --- |
| **class SV**  {  **private:**  char hk[30];  float dtb;  **public:**  **SV()**  {  }  **SV(char hk1[], float d)**  {  strcpy( hk, hk1);  dtb= d;  }  **friend istream & operator >>(istream &is, SV &p)**  {  is.ignore(1);  cout<<"\n Nhap hanh kiem: "; is.getline(p.hk,30);  cout<<"\n Nhap diem tb: "; is>>p.dtb;  return is;  }  **void Xuat()**  {  cout<<"\n Hanh Kiem: "<<hk;  cout<<"\n Diem TB: "<<dtb;  }  **char \*GetHk()**  {  return hk;  }  **float GetDtb()**  {  return dtb;  }  }; |

**Bước 3:** Viết lớp SVTC kế thừa từ hai lớp Nguoi, SV mô tả các đối tượng sinh viên tại chức.

|  |
| --- |
| **class SVTC : public Nguoi, public SV**  {  **private:**  float hp;  **public:**  **SVTC()**  {  }  // Ham tao lơp dan xuat goi lai ham tao hai lơp co so.  SVTC(char ht1[], int n1,char hk1[], float diem,  float hp1): Nguoi(ht1, n1),SV(hk1, diem)  {  hp=hp1;  }  **float hocbong()**  {  if(strcmpi(GetHk(),"Tot")==0 && GetDtb()>=9.0)  return 5000000;  else  if((strcmpi(GetHk(),"Tot")==0 && GetDtb()>=8.0)  || (strcmpi(GetHk(),"Kha")==0 && GetDtb()>=8.0))  return 3000000;  else  if((strcmpi(GetHk(),"Tot")==0 && GetDtb()>=7.0)  || (strcmpi(GetHk(),"Kha")==0 && GetDtb()>=7.0))  return 2000000;    return 0;  } |

- Định nghĩa toán tử nhập trong lớp dẫn xuất chú ý cần ép kiểu đối tượng lớp dẫn xuất để sử dụng toán tử nhập đã định nghĩa trong lớp cơ sở.

|  |
| --- |
| **friend istream & operator >>(istream &is, SVTC &p)**  {  is>>(Nguoi&) p; // goi lai toan tu >>  is>>(SV&) p;  cout<<"\n Hoc Phi: "; is>>p.hp;  }  **friend ostream & operator <<(ostream &os, SVTC p)**  {  p.Nguoi::Xuat();  p.SV::Xuat();  os<<"\n Hoc Phi: "<<p.hp;  if(p.hocbong()!=0)  os<<"\n Hoc bong: "<<p.hocbong() <<endl;  return os;  }  }; |

**Bước 4**: Viết hàm main() thực hiện yêu cầu đề bài.

|  |
| --- |
| **main()**  {  SVTC sv[30];  cout<<"\n Nhap Danh Sach Sinh Vien: "<<endl;  for( int i=0;i<2;i++)  {  cout<<"\n Nhap Sinh Vien Thu : "<<i+1<<endl;  cin>>sv[i];  }  cout<<"\n danh sach SVTC da nhap "<<endl;  for( int i=0;i<2;i++)  {  cout<<sv[i];  }  cout<<"\n\n Sinh vien duoc hoc bong"<<endl;  for( int i=0;i<2;i++)  if(sv[i].hocbong()!=0)  cout<<sv[i];  for( int i=0;i<1;i++)  for( int j=i+1; j<2;j++)  if(sv[i].GetDtb()> sv[j].GetDtb())  {  SVTC tg= sv[i];  sv[i]= sv[j];  sv[j]=tg;  }  cout<<"\n danh sach SVTC da sap "<<endl;  for( int i=0;i<2;i++)  cout<<sv[i];  } |

**Bài 3:**  Xây dựng một lớp đối tượng các máy in, lớp gồm có các thành phần:

- Thuộc tính: số hiệu và số l­ượng trong kho.

- Hàm nhapkho(int q) để nhập vào kho q đơn vị mặt hàng.

- Hàm xuatkho(int q) để xuất ra khỏi kho q đơn vị mặt hàng.

Xây dựng lớp máy in laser kế thừa từ lớp máy in và có thêm thuộc tính dpi

Xây dựng lớp máy in màu kế thừa từ lớp máy in có thêm thuộc tính băng màu.

Xây dựng lớp máy in laser màu kế thừa từ lớp máy in laser và máy in màu.

Viết chư­ơng trình quản lý các loại máy in với các thủ tục: nhập, xuất và in ra số lư­ợng các loại có trong kho.[3]

**Hướng dẫn:**

Bài thực hành cung cấp cho sinh viên kiến thức về cài đặt đa kế thừa. Cách khai báo cài đặt lớp cở sở ảo trong đa kế thừa.

Xây dựng sơ đồ lớp kế thừa như sau:

|  |
| --- |
| **MayIn** |
| # Mac: char[]  # Sluong: int  #MayIn() |
| + MayIn(char)  +void Nhap()  +void Xuat()  +void In() |

|  |
| --- |
| **MayInMau** |
| # BangMau: int |
| # MayInMau (int)  + MayInMau (char[], int)  +In(): void |

|  |
| --- |
| **MayInLaser** |
| # DPI: int |
| # MayInLaser(int)  + MayInLaser(char[], int)  + In():void |

|  |
| --- |
| **MayInLaserMau** |
|  |
| #MayInLaser(char, int, int )  +In(): void |

**- Bước 1:** Định nghĩa lơp MayIn

|  |
| --- |
| #include<iostream.h>  #include<string.h>  **class MayIn**  {  **protected:**  char Mac[25];  int Sluong;  //contructor chuan chi dung cho cac lop dan xuat  **MayIn()**{}  **public:**  **MayIn:: MayIn( char\*s): Sluong(0)**  {  strcpy( Mac, s);  }  **void Nhap( int sl)**  {  Sluong += sl;  }  **void xuat(int sl)**  {  Sluong = (Sluong > sl) ? Sluong-sl : 0;  }  **void In()**  {  cout<<" May In "<< Mac <<" ton "<< Sluong<<" chiec\n";  }  }; |

Nhận thấy lớp MayInLaser và lớp MayInMau cùng kế thừa từ lớp MayIn. Lớp MayInLaserMau tiếp tục kế thừa từ hai lớp MayInLaser và lớp MayInMau. Do đó, các thành phần dữ liệu kế thừa từ lớp MayIn sẽ bị lặp lại trong các đối tượng của lớp MayInLaserMau. Khắc phục điều này cần sử dụng từ khóa virtual đặt trước mỗi chỉ thị kế thừa lớp MayIn xuống hai lớp MayInLaser và lớp MayInMau. Lớp MayIn sẽ là lớp cơ sở ảo của hai lớp MayInLaser và lớp MayInMau.

* **Bước 2:** Định nghĩa lớp MayInLaser.

|  |
| --- |
| // khai bao lop co so ao  **class MayInLaser: public virtual MayIn** {  **protected:**  int DPI;  **MayInLaser( int dpi):DPI(dpi)** {}  **public:**  **MayInLaser(char\*s,int dpi):MayIn(s),DPI(dpi)** {}  **void In()**  {  cout<<" May in laser"<<Mac<<"("<<DPI<<"dpi) ton"  <<Sluong<<"chiec\n";  }  }; |

* **Bước 3 :** Định nghĩa lớp MayInMau.

|  |
| --- |
| **class MayInMau: public virtual MayIn**  {  **protected**:  int BangMau;  **MayInMau(int bm): BangMau(bm)** {}  **public:**  **MayInMau(char\*s,int bm): MayIn(s), BangMau (bm)**  {  }  **void In()**  {  cout<<"Mau in mau"<<Mac<<"("<<BangMau<<"bang mau) ton";  cout<<Sluong<<" chiec\n";  }  }; |

* **Bước 4: Định nghĩa lớp MayInLaserMau**

|  |
| --- |
| **class MayInLaserMau: public MayInLaser, public MayInMau**  {  **public**:  **MayInLaserMau(char\*s, int dpi, int bm):**  **MayIn(s),MayInLaser(dpi),MayInMau(bm)**  { }  **void In()**  {  cout<<" May in laser mau "<<Mac<<" ("<<DPI<<" dpi,"  <<BangMau<<"bang mau) ton "<<Sluong<<" chiec\n";  }  }; |

* **Bước 5: Định nghĩa hàm main()**

|  |
| --- |
| main()  {  MayIn s1("Epson");  MayInLaser s2("HP6",600);  MayInMau s3("Apple",3);  MayInLaserMau s4("Kodak",400,3);  s1.Nhap(5);  s2.Nhap(7);  s1.xuat(2);  s3.Nhap(2);  s2.xuat(3);  s4.Nhap(6);  s1.In();  s2.In();  s3.In();  s4.In();  } |

**Bài tập tự giải**

**Bài 4:** Xây dựng lớp sinh viên (STUDENT) gồm các thuộc tính: Mã sinh viên, họ tên, giới tính. Lớp bổ sung thêm các hàm thành phần:

- Hàm tạo, hàm hủy.

- Định nghĩa toán tử nhập thông tin một sinh viên.

- Định nghĩa toán tử xuất thông tin một sinh viên.

Xây dựng lớp môn học (MONHOC) gồm các thuộc tính:mã môn, tên môn. Lớp này bổ sung thêm các hàm thành phần:

- Hàm tạo, hàm hủy

- Định nghĩa toán tử nhâp >>, xuất <<

Xây dựng lớp điểm môn học(DIEMMH) kế thừa từ hai lớp trên và bổ sung thêm thuộc tính: điểm thi.

Viết chương trình thực hiện yêu cầu sau:

- Nhập vào danh sách quản lý các đối tượng điểm môn học.

- Hiển thị danh sách đã nhập.

- Liệt kê sinh viên giới tính là ‘nam” trong danh sách.

- Tính trung bình điểm của các môn học của danh sách điểm môn học

- Tìm đối tượng điểm môn học có tên sinh viên là X (X nhập vào từ bàn phím).

- Tìm thông tin của điểm môn học có tên môn là “Toan” điểm thi nhỏ nhất..

- Sắp xếp danh sách điểm môn học theo chiều giảm của điểm thi.

**Hướng dẫn:**

Xây dựng chương trình theo sơ đồ kế thừa sau:

**STUDENT**

**MONHOC**

**DIEMMH**

Bước 1: Xây dựng lớp STUDENT

Bước 2: Xây dựng lớp MONHOC

Bước 3: Xây dựng lớp DIEMMH kế thừa từ hai lớp STUDENT và MONHOC

Bước 4: Viết hàm main() thực hiện các chức năng trong chương trình.

**Bài 5:** Xây dựng một lớp Person mô tả các thông tin về ngư­ời. Lớp gồm các thuộc tính:

- Name: Mô tả họ và tên của ngư­ời, là một xâu không quá 50 ký tự.

- Address: Mô tả địa chỉ của ng­ời, là một xâu không quá 40 ký tự.

- Phone: Mô tả số điện thoại, là một số nguyên.

- Hàm tạo không tham số.

- Hàm tạo 3 tham số.

Xây dựng một lớp Officer mô tả các thông về một cán bộ viên chức, lớp kế thừa từ lớp Person và bổ sung thêm các thành phần:

- Salary: Mô tả lương của cán bộ, là một số thực.

- Hàm tạo không tham số.

- Hàm tạo 4 tham số.

Xây dựng một lớp Student mô tả các thông tin về một sinh viên, lớp kế thừa từ lớp Person và bổ sung thêm các thành phần:

- Fee: Mô tả học phí của sinh viên, là một số thực.

- Hàm tạo không tham số.

- Hàm tạo 4 tham số.

Xây dựng một lớp OffStudent kế thừa từ hai lớp Officer và Student, lớp bổ sung thêm các thành phấn sau:

- Hàm tạo 5 tham số.

- Hàm OutScreen() để in thông tin về một đối tượng OffStudent ra màn hình.

Viết chư­ơng trình khai báo một mảng đối tượng OffStudent, nhập dữ liệu vào để tạo danh sách đối tượng OffStudent. In dữ liệu ra màn hình. Tìm đối tượng có lương là 5 triệu. Sắp xếp danh sách các đối tượng theo chiều tăng dần của tên.