# B**ÀI THỰC TẬP 06 - LẬP TRÌNH CÀI ĐẶT ĐA HÌNH**

## A. MỤC TIÊU

-  Hiểu khái niệm về đa hình.

- Biết cách sử dụng phương thức ảo, phương thức thuần ảo.

- Biết cách thiết kế các lớp và vận dụng tính đa hính để xây dựng chương trình một cách hệ thống.

## B. NỘI DUNG

- Khái niệm đa hình

- Phương thức ảo, phương thức thuần ảo

- Lớp cơ sở trừu tượng.

## 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 kế thừa. Vận dụng tính đa hình xây dựng chương trình một cách tổng quát dễ phát triển và nâng cấp.

## E. TÓM TẮT LÝ THUYẾT

1. **Phương thức ảo**

Phương thức ảo là một hàm thành viên khai báo trong lớp cơ sở và được định nghĩa lại (ghi đè) trong các lớp dẫn xuất.

Khai báo phương thức ảo bằng cách thêm từ khóa virtual ở phía trước. Khi đó các phương thức có cùng kiểu, cùng tên và cùng danh sách tham số với phương thức này trong các lớp dẫn xuất cũng là phương thức ảo.

Cú pháp:

|  |
| --- |
| virtual Kiểu\_dữ\_liệu Tên\_phương\_thức (Danh sách tham số){  } |

**Các lưu ý khi sử dụng phương thức ảo:**

Phương thức ảo sẽ cung cấp giải pháp (tại thời điểm thực thi):

* Tham chiếu đến kiểu thực sự của đối tượng.
* Sự kết nối xảy ra tại thời điểm thực thi là kết nối động.
* Làm giảm tốc độ thực thi của chương trình
* Chỉ hoạt động thông qua con trỏ.
* Chỉ hoạt động nếu các hàm ở lớp cơ sở và lớp con có nghi thức giao tiếp giống hệt nhau (cùng tên, cùng kiểu, cùng danh sách tham số)
* Nếu ở lớp con không định nghĩa lại phương thức ảo thì sẽ gọi phương thức ở lớp cơ sở (gần nhất có định nghĩa).

1. **Phương thức thuần ảo**

Phương thức thuần ảo là một phương thức ảo mà nội dung của nó không có gì.

Cú pháp:

|  |
| --- |
| virtual Kiểu\_dữ\_liệu Tên\_phương\_thức (Danh sách tham số) = 0; |

Một phương thức thuần ảo bắt buộc phải được ghi đè lại trong lớp dẫn xuất kế thừa từ lớp chứa phương thức thuần ảo này.

1. **Lớp cơ sở trừu tượng**

Lớp chứa ít nhất một phương thức ảo thuần túy được gọi là lớp cơ sở trừu tượng (asbtract class). Một lớp cơ sở trừu tượng là một lớp chỉ được dùng làm cơ sở, định nghĩa một số khái niệm tổng quát, chung cho các lớp khác. Lớp trừu tượng tồn tại vì mục đích sử dụng để các lớp khác kế thừa từ nó do đó không có đối tượng thực sự của lớp này được tạo ra trong chương trình.

|  |
| --- |
| class Box  {  public:  virtual double TheTich()=0;  private:  double chieudai, chieurong, chieucao;  } |

## F. HƯỚNG DẪN CHI TIẾT

**Bài 1:** Cho lớp độc giả (DocGia) mô tả các đối tượng độc giả. Lớp có thuộc tính:mã, họ, tên, số tháng hiệu lực. Lớp độc giả trẻ (DGTre), lớp độc giả người lớn (DGLon) kế thừa từ lớp độc giả. Lớp đọc giả trẻ bổ sung thêm thuộc tính: năm sinh. Lớp đôc giả người lớn bổ sung thêm thuộc tính: nghề nghiệp.

- Quản lý danh sách các loại độc giả (cả hai loại).

- Tìm độc giả trẻ em có số tháng hiệu lực <10, độc giả người lớn là giảng viên.

- Tính tổng tiền làm thẻ của tất cả các độc giả. Biết độc giả trẻ em tiền làm thẻ = số tháng hiệu lực \* 20000, độc giả người lớn tiền làm thẻ = số tháng hiệu lực \* 30000

**Hướng dẫn:**

Bài thực hành hướng dẫn sinh viên cách cài đặt phương thức ảo, vận dụng thể hiện tính đa hình viết chương trình.

Xây dựng chương trình theo sơ đồ kế thừa sau:

|  |
| --- |
| **Docgia** |
| - ma: char[]  - ho, ten : char[]  - sthl: int |
| + nhap(): void  + xuat(): void  + virtual TienThe()=0: float  + virtual Tim(): int |

|  |
| --- |
| **DGLon** |
| - nghe[]: char |
| + nhap(): void  + xuat(): void  + virtual TienThe(): float  + virtual Tim():int |

|  |
| --- |
| **DGTre** |
| - ns: int |
| + nhap(): void  + xuat(): void  + virtual TienThe(): float  + virtual Tim(): int |

**- Bước 1**: Xây dựng lớp DocGia mô tả thông tin độc giả.

+ Định nghĩa hai phương thức ảo: Nhap(), Xuat()

+ Định nghĩa hai phương thức thuần ảo: TienThe(), Tim()

Phương thức ảo và thuần ảo cần bổ sung từ khóa virtual phía trước.

|  |
| --- |
| **#include<iostream.h>**  **class DocGia**  {  **protected:**  char ma [10], ho[30], ten[30];  int sthl;  **public:**  virtual void Nhap()  {  cout<<"\n Nhap ma: "; cin.getline(ma,30);  cout<<"\n Nhap ho: "; cin.getline(ho,30);  cout<<"\n Nhap ten: "; cin.getline(ten,30);  cout<<"\n Nhap sthl: "; cin>>sthl;  }  **virtual void Xuat()**  {  cout<<"\n Ma: "<<ma;  cout<<"\n Ho"<<ho;  cout<<"\n Ten: "<<ten;  cout<<"\n So thang hieu luc cua the: "<<sthl ;  }  **virtual float TienThe()=0 // ham thuan ao**  **virtual int Tim() // ham ao**  }; |

**- Bước 2**: Xây dựng lớp độc giả trẻ em. Trong lớp này ghi đè lại các phương thức ảo và thuần ảo đã định nghĩa ở lớp độc giả. Chú ý: có thể có hoặc không nhắc lại từ khóa virtual ở các hàm được ghi đè trogn lớp dẫn xuất.

|  |
| --- |
| **class DGTre: public DocGia**  {  int ns;  **public:**  **void Nhap()**  {  cin.ignore();  DocGia:: Nhap();  cin.ignore();  cout<<" \n Nhap ns: "; cin>>ns;  }  **virtual void Xuat()**  {  DocGia:: Xuat();  cout<<"\n Nam sinh: "<<ns;  }  **float TienThe()**  {  return sthl \* 200000;  }  **int Tim()**  { if(sthl<10) return 1;    return 0;  }  }; |

**- Bước 3:** Tương tự xây dựng lớp độc giả người lớn kế thừa từ lớp độc giả

|  |
| --- |
| **class DGLon: public DocGia**  {  char nghe[20];  **public:**  **virtual void Nhap()**  {  cin.ignore();  DocGia:: Nhap();  cin.ignore();  cout<<" \n Nhap nghe Nghiep: ";cin.getline(nghe,30);  }  **virtual void Xuat()**  {  DocGia:: Xuat();  cout<<"\n Nghe Nghiep: "<<nghe;  }  **float TienThe()**  {  return sthl \* 300000;  }  **int Tim()**  {  if(strcmp( nghe, "Giang vien")==0)  return 1;  return 0;  }  }; |

**- Bước 4**: Xây lớp quản lý độc giả để quản lý các loại độc giả.

- Khai báo con trỏ lớp độc giả làm thành phần thuộc tính.

- Các hàm thành phần:

+ Nhap(): Cho phép lựa chọn đối tượng cần quản lý qua từng vòng lặp bằng cách sử dụng toán tử new cấp phát động. Nếu chọn 1 tạo mới đối tượng độc giả trẻ em, chọn 2 tạo mới đối tượng độc giả người lớn. Tiếp đó, con trỏ d[i] gọi đến hàm Nhap() là phương thức ảo để nhập dữ liệu, tùy vào đối tượng d[i] đang quản lý là đối tượng nào nó sẽ gọi hàm Nhap() tương ứng của lớp độc giả trẻ em hay độc giả người lớn. Sau khi mỗi đối tượng độc giả được nhập dữ liệu tiếp tục gọi đến hàm TienThe() để tính tiền làm thẻ của mỗi đối tượng và cộng vào biến s. Kết thúc vòng lặp tổng tiền làm thẻ được lưu trong s.

+ TimKiem(): Tìm các loại độc giả trẻ em có số tháng hiệu lực <10, độc giả người lớn là giảng viên bằng cách gọi đến phương thức Tim() là phương thức ảo trong mỗi đối tượng độc giả.

|  |
| --- |
| **class QuanLyDG**  {  **public:**  DocGia \*d[30];  int n ;  **void Nhap()**  {  int chon, i=0; float s=0 ;  while(1)  {  cout<<"\n\*DocgiaTreEm /DocGiaNguoiLon/Dung (1,2,3):";  cin>>chon;  if (chon==3) break;  if (chon==1) d[i]=new DGTre();  if (chon==2) d[i]=new DGLon();  d[i]->Nhap();  s= s + d[i]->TienThe();  i++;  n=i;  }  cout<<"\n Tong Tien Lam The :"<<s;  }  **void TimKiem()**  {  cout<<"\n Danh sach doc gia can tim: "<<endl;  for(int j=0; j<n;j++)  if(d[j]->Tim()==1)  {  d[j]->Xuat();  cout<<"\n------------------------"<<endl;  }  }  }; |

**- Bước 5.** Viết hàm main() tạo danh sách độc giả và thực hiện yêu cầu đề bài.

|  |
| --- |
| **main()**  {  QuanLyDG a;  cout<<"\n CHUONG TRINH QUAN LY DOC GIA"<<endl;  a.Nhap();  a.TimKiem();  } |

**Bài 2:** Cho lớp người (Nguoi) mô tả các đối tượng người. Lớp có thuộc tính: họ và tên. Lớp sinh viên (Sinhvien), lớp giảng viên (Giangvien) kế thừa từ lớp người. Lớp sinh viên bổ sung thêm thuộc tính: điểm thi. Lớp giảng viên bổ sung thêm thuộc tính: số bài báo.

- Tạo danh sách các đối tượng sinh viên và giảng viên (danh sách chứa hai loại đối tượng).

- Hiển thị danh sách đã nhập.

- Tìm đối tượng được thưởng. Biết sinh viên được thưởng nếu điểm thi lớn hơn 8, giảng viên được thưởng nếu số bài báo lớn hơn 5.

**Hướng dẫn chi tiết:**

Xây dựng các lớp theo sơ đồ kế thừa như sau:

|  |
| --- |
| Nguoi |
| - hoten[]: char[] |
| + nhap(): void  + thuong(): int  + xuat(): void |

|  |
| --- |
| Sinhvien |
| - dttn: float |
| + nhap(): void  + thuong(): int  + xuat():void |

|  |
| --- |
| Giangvien |
| - sobb: int |
| + nhap():void  + thuong(): int  + xuat(): void |

**- Bước 1:** Tạo lớp Nguoi mô tả thông tin các đối tượng người. Trong lớp mô tả các thành phần:

- Thuộc tính: hoten;

- Phương thức ảo: nhap(), xuat(), thuong().

|  |
| --- |
| **#include <iostream.h>**  **#include <conio.h>**  **class Nguoi**  {  char hoten[30];  public:  **virtual void nhap()**  {  cout<<"\nHo ten:"; cin.getline(hoten,30);  }  **virtual int thuong()**  {  }  **virtual void xuat()**  {  cout<<"\nHo ten:"<<hoten;  }  }; |

**- Bước 2:** Xây dựng lớp Sinhvien kế thừa từ lớp Nguoi

Trong lớp Sinhvien ghi đè lại các phương thức ảo nhap(), xuat(), thuong() ở lớp Nguoi. Không nhất thiết phải nhắc lại từ khóa virtual trước các phương thức này.

|  |
| --- |
| **class Sinhvien:public Nguoi**  {  float dttn;  **public:**  **void nhap()** // ghi de phuong thuc  {  Nguoi::nhap();  cout<<"\nDiem thi tn:"; cin>>dttn;  }  **int thuong()** // ghi de phuong thuc  {  return (dttn>8?1:0);  }  **void xuat()** // ghi de phuong thuc  {  cout<<"\n-Sinh vien:";  Nguoi::xuat();  cout<<"\nDiem thi tn:"<<dttn;  }  }; |

**- Bước 3:** Tương tự xây dựng lớp Giangvien kế thừa từ lớp Nguoi, tương tự như lớp Sinhvien

**- Bước 4:** Xây dựng hàm main()

Khai báo con trỏ lớp Ngươi

|  |
| --- |
| Nguoi \*ds[100]; |

Sử dụng cấp phát động và tạo menu lựa chọn đối tượng được quản lý qua từng lượt chạy. Nếu chọn 1 con trỏ ds[k] sẽ quản lý đối tượng giảng viên, chọn 2 sẽ quản lý đối tượng sinh viên.

|  |
| --- |
| while(1)  {  cout<<"\n\*Gv/Sv/Thoat (1,2,3):";  cin>>chon; cin.get();  if (chon==1) ds[k]=new Giangvien();  if (chon==2) ds[k]=new Sinhvien();  if (chon==3) break;  ds[k]->nhap(); k++;  } |

- Xuất danh sách đối tượng được thưởng

|  |
| --- |
| cout<<"\n\*Danh sach nhung nguoi duoc thuong: ";  for (i=0; i<k; i++)  if (ds[i]->thuong()) ds[i]->xuat(); |

- Hàm main() hoàn chỉnh.

|  |
| --- |
| **main()**  {  Nguoi \*ds[100]; int k=0, chon, i;  while(1)  {  cout<<"\n Moi lua chon nhap doi tuong"<<endl;  cout<<"\n\*Gv/Sv/Thoat (1,2,3):";  cin>>chon; cin.get();  if (chon==1) ds[k]=new Giangvien();  if (chon==2) ds[k]=new Sinhvien();  if (chon==3) break;  ds[k]->nhap(); k++;  }  cout<<"\n\*Danh sach nhung nguoi duoc thuong: ";  for (i=0; i<k; i++)  if (ds[i]->thuong())  ds[i]->xuat();  } |

**Bài 3**: Xây dựng lớp lớp SHAPE mô tả hình. Các lớp hình tròn ROUND, hình chữ nhật REC, hình tam giác TRI kế thừa từ lớp SHAPE. Lớp ROUND bổ sung thêm bán kính, lớp REC bổ sung thêm chiều dài và rộng, lớp TRI bổ sung thêm ba cạnh của tam giác. - Tạo một mảng chứa danh sách các loại hình. Tính tổng chu vi,diện tích của tất cả các hình đã nhập.[1]

**Hướng dẫn:** Xây dựng các lớp theo sơ đồ kế thừa sau:

**TRI**

**SHAPE**

**ROUND**

**REC**

**- Bước 1:** Xây dựng lớp SHAPE quản lý các hình. Trong lớp định nghĩa các hàm thuần ảo nhap(), xuat(), dtich(), chuvi().

|  |
| --- |
| virtual void nhap()=0;  virtual void xuat()=0;  virtual float dtich()=0;  virtual float chuvi()=0;  friend float sumdt();  friend float sumcv(); |

**- Bước 2:** Viết lớp ROUND kế thừa từ lớp SHAPE

|  |
| --- |
| **class ROUND:public SHAPE**  {  **private**:  float r;  **public:**  **ROUND(float r=0)**  {  this->r=r;  }  **void nhap()**  {  cout<<"Nhap ban kinh: “;  cin>>r;  }  **void xuat()**  {  cout<<"Hinh tron co ban kinh la: "<<r<<endl;  cout<<"\n Chu vi: "<<chuvi()<<endl;  cout<<"\n Dien tich: "<<dtich();  }  **float dtich()**  {  return 3.14\*r\*r;  }  **float chuvi()**  {  return 2\*3.14\*r;  }  }; |

**- Bước 3:** Tương tựviết lớp TRI kế thừa từ lớp SHAPE

**- Bước 4:** Tương tự viết lớp REC kế thừa từ lớp SHAPE

**- Bước 5:** Định nghĩa hai hàm sumdt(), sumcv() là hàm bạn của lớp SHAPE có chức năng tính tổng diện tích, tổng chu vi của các hình.

|  |
| --- |
| **void sumdt(SHAPE \*a[],int n)**  {  float sum1=0;  for(int i=0;i<n;i++)  {  sum1+=a[i]->dtich();  }  cout<<"\nTong dien tich la: "<<sum1<<endl;  }  **void sumcv(SHAPE \*a[],int n)**  {  float sum2=0;  for(int i=0;i<n;i++)  {  sum2+=a[i]->chuvi();  }  cout<<"Tong chu vi la: "<<sum2<<endl;  } |

**- Bước 6:** Viết hàm main() thực thi chương trình

|  |
| --- |
| **int main()**  {  SHAPE \*s[100];  cout<<"Nhap so luong hinh: ";  int n;  cin>>n;  int i=0;  for(int i=0;i<n;i++){  cout<<"1.Nhap hinh tron"<<endl;  cout<<"2.Nhap hcn"<<endl;  cout<<"3.NHap tam giac"<<endl;  cout<<"4.Exit"<<endl;  int c;  cout<<"\n => Moi Nhap lua chon: ";cin>>c;cin>>c;  if(c==1)  {  s[i]=new ROUND();  }  else if(c==2)  {  s[i]=new REC();  }  else if(c==3)  {  s[i]=new TRI();  }  s[i]->nhap();  }  for(int i=0;i<n;i++)  {  cout<<"\n- Thong tin hinh thu: "<<i+1<<endl;  s[i]->xuat();  }  sumdt(s,n);  sumcv(s,n);  } |

**Bài 4:** Tương tự bài 3 sinh viên tự thiết kế các lớp như trong sơ đồ sau. Chương trình cho phép in ra thông tin gồm thành phần dữ liệu của mỗi loại hình, chu vi và diện tích của nhóm hình hai chiều, thể tích của nhóm hình ba chiều của các hình.[1]

**Vuong**

**Hinhve**

**Haichieu**

**Bachieu**

**Tron**

**Chunhat**

**Cau**

**Lapphuong**

**Bài 5:** Một nhà cung cấp dịch vụ Internet (*ISP*) có khả năng cung cấp 3 loại hình thuê bao Internet sau:

* Dial Up: thuê bao hằng tháng là 30.000 VNĐ và khách hàng trả tiền theo khoảng thời gian truy cập với đơn giá là 30 VNĐ/phút.
* ADSL: thuê bao hằng tháng là 50.000 VNĐ và khách hàng trả tiền theo dung lượng dữ liệu truyền tải với đơn giá là 50 VNĐ/MB.
* T1: khách hàng trả một khoảng tiền thuê bao hằng tháng cố định là 2.000.000 VNĐ.

Khách hàng có thể chọn một trong 3 loại hình thuê bao Internet trên. Vào cuỗi mỗi tháng, nhân viên tính cước phải trình báo lên ban giám đốc tổng số tiền cước thu được của tất cả các thuê bao. Vì đây là nhà cung cấp lớn nên nhân viên tính cước phải mất rất nhiều thời gian và công sức để làm việc này.

Viết chương trình thực hiện các công việc sau:

* Xây dựng các lớp và các phương thức cần thiết.
* Nhập vào thông tin của từng thuê bao trong tháng.
* Tính tổng tiền cước của từng loại thuê bao.
* Tính tổng số tiền cước thu được của tất cả các thuê bao trong tháng.
* Tìm và xuất thông tin thuê bao ADSL có tiền cước lớn nhất.

**Bài 6:** Xây dựng chương trình quản lý danh sách các giao dịch nhà đất. Thông tin bao gồm:

▪ Giao dịch đất: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), đơn giá, loại đất (loại A, B, C), diện tích.

- Nếu là loại B, C thì: thành tiền = diện tích \* đơn giá.

- Nếu là loại A thì: thành tiền = diện tích \* đơn giá \* 1.5

▪ Giao dịch nhà: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), đơn giá, loại nhà (cao cấp, thường), địa chỉ, diện tích.

- Nếu là loại nhà cao cấp thì: thành tiền = diện tích \* đơn giá.

* Nếu là loại thường thì: thành tiền = diện tích \* đơn giá \* 90%

**Thực hiện các yêu cầu sau:**

▪ Nhập xuất danh sách các giao dịch.

▪ Tính tổng số lượng cho từng loại giao dịch

▪ Tính trung bình thành tiền của giao dịch đất.

▪ Xuất ra các giao dịch của tháng 9 năm 2013.