1 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

|  |
| --- |
| **BÀI 5: TÍNH ĐA HÌNH : Polymorphism** |

**1. Khái niệm**

Phương thức của lớp cha khi thực hiện sẽ được thay thế bằng một phương thức của lớp con thì phương thức  này gọi là có tính đa hình. Tính đa hình giúp cho việc lập trình đơn giản và dễ mở rộng. Để cài đặt phương  thức có tính đa hình ta dùng phương thức ảo và phương thức thuần ảo.

**2. Phương thức ảo (virtual method)**

Phương thức ảo là phương thức được định nghĩa ở lớp cơ sở (lớp cha) mà các lớp dẫn xuất (lớp con) muốn  sử dụng phải định nghĩa lại. Dùng từ khoá virtual để khai báo phương thức ảo:

|  |
| --- |
| **virtual <kiểu trả về> <tên phương thức >(<d/s tham số>)** |

**{…}**

Phương thức khởi tạo không được là phương thức ảo nhưng phương thức hủy bỏ có thể là phương thức ảo.  Dùng phương thức ảo chậm hơn phương thức thông thường vì khi thực hiện mới được xác định cụ thể. **Ví dụ:**

#include <iostream>

using namespace std;

class A

{ public:

virtual void hello() {cout<<" A Hello"<<endl;}

};

class B: public A

{ public:

void hello() { cout<<" B Hello"<<endl; }

};

class C: public A

{ public:

void hello() { cout<<" C Hello"<<endl; }

};

int main()

{ A \*p = new A; p->hello();

B b; p=&b; p->hello();

C c; p=&c; p->hello();

cout<<" TRUONG HOP KHONG SU DUNG CON TRO"<<endl;

A x; x.hello();

B y; y.hello();

C z; z.hello();

}

|  |
| --- |
| **Cài đặt 3 files theo 2 trường hợp có virtual và KHÔNG có virtual + thừa kế của bài minh họa trên:  chạy thử**  **VIẾT BỔ SUNG HÀM**  **+ Hàm thiết lập (có tham số mặc định)**  **+ NHẬP() /XUẤT()**  **+ đầy đủ các hàm set() và get() cho class A, B, C (B, C con thừa kế A)**  + Thực nghiệm trường hợp ngược lại không được:   con trỏ của đối tượng lớp con trỏ vào đ/c của đối tượng thuộc lớp cha:  KHÔNG ĐƯỢC A a;   B \*b = new B; b = &a; : SAI |

2 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

**Nhận xét:**

- Phương thức chao() có tính đa hình: cùng lời gọi pa->chao() nhưng lần 1 gọi chao cua A, lan 2 gọi  chao cua B, lan 3 goi chao cua C.

- Nếu trong lớp B, C không định nghĩa lại phương thức chào thì cả ba lần đều gọi chào của A. - Nếu phương thức chao() trong lớp A không khai báo virtual thì phương thức chao() sẽ không có tính  đa hình, khi đó cả ba lần đều gọi chào của A.

- Có thể gán địa chỉ của đt thuộc lớp con (&b, &c) vào biến con trỏ, trỏ tới đt thuộc lớp cha (\*pa) nhưng *không thể làm ngược lại* (áp dụng nguyên tắc **“con gán vào cha”** đối với biến kiểu đối tượng  hoặc biến kiểu con trỏ, trỏ tới đối tượng)

**3. Phương thức thuần ảo, lớp trừu tượng**

Phương thức thuần ảo là phương thức ảo nhưng không có lệnh (phương thức rỗng). Phương thức thuần ảo  có dạng:

|  |
| --- |
| **virtual <kiểu trả về> <tên phương thức >(<d/s tham số>) = 0;** |

\* Lớp có phương thức thuần ảo gọi là lớp trừu tượng (abstract class).

VD: class A <= lớp đt “trưu tượng”, vì trong nó có “phương thức thuần ảo” là f() {…

 public:

 virtual void f() = 0; // “phương thức thuần ảo”

 …

}

\* Nếu một lớp thừa kế lớp trừu tượng mà không định nghĩa lại phương thức thuần ảo thì lớp thừa kế cũng  là lớp trừu tượng.

VD: class B: public A <= class B cũng là “lớp đối tượng TRỪU TƯỢNG“ luôn, mặc dù ko có virtual  {

 ….(không định nghĩa lại :phương thức thuần ảo” f()

 }

=====> *Lưu ý là không được (tạo) khai báo đối tượng thuộc lớp trừu tượng*.

VD: A a; <= SAI

 A \*p; <= SAI

 B b; <= SAI

 B \*q; <= SAI

class C: public B <= class C không phải class TRỪU TƯỢNG, vì đã đinh nghĩa “phương thức” f() { …

 public:

 void f() { …..}

 ….

}

…

C c; <= ĐÚNG

C \*d; <= ĐÚNG

========================

==========================

Thường ta chọn phương thức ở lớp cha, mà chưa thể xác định cách thực hiện, làm phương thức thuần ảo. Ở  lớp con ta sẽ định nghĩa lại phương thức thuần ảo, để xác định cụ thể cách thức thực hiện.

3 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

**Ví dụ:**

Nhập một danh sách gồm giảng viên và sinh viên, in ra danh sách những người được thưởng. Biết rằng điều  kiện được thưởng là giảng viên có số bài báo >3, sinh vien có điểm thi tốt nghiệp >8. #include <iostream>

using namespace std;

class person

{ char name[30];

public: virtual void nhap() {cout<<" Nhap ho ten:"; cin.ignore(); cin.getline(name,30); } virtual int thuong() = 0;

virtual void xuat() {cout<<name; }

};

class lecturer: public person

{ int tno;

 public: void nhap(){ cout<<" NHAP THONG TIN GIANG VIEN:"<<endl;

 person::nhap();

 cout<<" Nhap so bai bao:"; cin>>tno;

 }

 int thuong() { return (tno>3? 1: 0); }

 void xuat(){ cout<<" GIANG VIEN: "; person::xuat(); cout<<" so bai bao = "<<tno<<endl; } };

class student: public person

{ float gra;

 public: void nhap(){cout<<" NHAP THONG TIN SINH VIEN:"<<endl;

 person::nhap();

 cout<<" Nhap diem tot nghiep:"; cin>>gra;

 }

 int thuong(){return(gra > 8.0? 1: 0); }

 void xuat(){cout<<" SINH VIEN: "; person::xuat(); cout<<" diem tot nghiep ="<<gra<<endl;  };

int main()

{ person \*list[100];

int ch; //nhap lua chon

int n = 0; //thu tu nguoi (GV or SV) duoc nhap

while(1) // nhap cho den khi NSD tu ket thuc = while(TRUE)

{ cout<<"CHON MOT TRONG CAC TRUONG HOP SAU:"<<endl;

cout<<"1. NHAP GIANG VIEN "<<endl;

cout<<"2. NHAP SINH VIEN "<<endl;

cout<<"so khac: THOAT "<<endl;

cout<<"Nhap so lua chon: "; cin>>ch;

if(ch != 1 && ch !=2 ) break; // Ket thuc nhap = ket thuc lenh while(TRUE)

if(ch == 1) list[n] = new lecturer; //khai báo phan tu moi cua ds là GV, vi da chon nhap GV if(ch == 2) list[n] = new student; //khai báo phan tu moi cua ds là SV, vi da chon nhap SV list[n]->nhap(); //thuc hien thu tuc nhap

n++; //nhap person ke tiep

}

cout<<" NHUNG NGUOI SAU DAY LA DUOC THUONG:"<<endl;

for(int i = 0; i<n; i++) if(list[i]->thuong()) list[i]->xuat();

}

|  |
| --- |
| **Cài đặt thêm bài minh họa trên: chạy thử**  + Hãy mở rộng bài tập trên bằng cách thêm một lớp nhân viên,   biết rằng nhân viên có XẾP LOẠI LAO ĐỘNG LÀ ‘A’ là được thưởng.  **+VIẾT BỔ SUNG HÀM**  **. Hàm thiết lập (có tham số mặc định)**  **. NHẬP() /XUẤT()**  **. đầy đủ các hàm set() và get() cho tất cả class** |

4 TS. GVC. VÕ XUÂN THỂ: voxuanthephd@gmail.com (công việc) tranhailua@yahoo.com (cá nhân)

CHÚ Ý: dùng: #include <ctype.h>

CÓ THỂ chuyển KÝ TỰ sang “KÝ TỰ HOA” : c = toupper(c);

CÓ THỂ chuyển KÝ TỰ sang “ký tự thường : c = tolower(c);

**Nhận xét:**

Do phương thức “nhập, thưởng, xuất” là phương thức ảo của lớp người (lớp cha) nên các phương thức này  khi thực thi sẽ có tính đa hình: có khi gọi “nhập, thưởng, xuất” của lớp giảng viên (lớp con), có khi thì gọi  “nhập, thưởng, xuất” của lớp sinh viên (lớp con) tuỳ theo con trỏ ds[i] đang giữ địa chỉ của đối tượng giảng  viên hay sinh viên.