Chương 11
ĐA XẠ – ĐA HÌNH – POLYMORPHISM

- Nguyễn Hữu Lợi
- Đoàn Chánh Thống
- ThS. Nguyễn Thành Hiệp
- ThS. Trương Quốc Dũng
- ThS. Võ Duy Nguyên
- ThS. Nguyễn Văn Toàn
- TS. Nguyễn Duy Khánh
- TS. Nguyễn Tấn Trần Minh Khang

1. MỤC TIÊU

Mục tiêu

- Hiểu được cơ chế hoạt động của phương thức ảo.
- Ứng dụng được phương thức ảo.
- Thi cao học đề thi hay hỏi phần này.
- Phỏng vấn xin việc người ta cũng rất hay hỏi.

2. VÍ DỤ DẪN NHẬP

Ví dụ dẫn nhập

- Hãy thực hiện chương trình dưới đây và cho biết kết quả của việc chạy chương trình trong bốn trường hợp:
- Trường hợp 1: **XXXX** là khoảng trắng,
  YYYY là khoảng trắng.
- Trường hợp 2: **XXXX** là `virtual`,
  YYYY là khoảng trắng.
- Trường hợp 3: **XXXX** là khoảng trắng,
  YYYY là `virtual`.
- Trường hợp 4: **XXXX** là `virtual`,
  YYYY là `virtual`.

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.    public:
15.        XXXX void Sketchy()
16.        {
17.            cout<<"\n A's Sketchy()";
18.            Sketchy(-1);
19.        }
20.        YYYY void Sketchy(int num)
21.        {
22.            cout<<"\n A's Sketchy("<<num<<")";
23.        }
24.};
```

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.    public:
15.        XXXX void Sketchy()
16.        {
17.            cout<<"\n A's Sketchy()";
18.            Sketchy(-1);
19.        }
20.        YYYY void Sketchy(int num)
21.        {
22.            cout<<"\n A's Sketchy("<<num<<")";
23.        }
24.};
```

Khai báo lớp đối tượng A lớn

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.    public:
15.        XXXX void Sketchy()
16.        {
17.            cout<<"\n A's Sketchy()";
18.            Sketchy(-1);
19.        }
20.        YYYY void Sketchy(int num)
21.        {
22.            cout<<"\n A's Sketchy("<<num<<")";
23.        }
24.};
```

Phương thức Sketchy() được khai báo trong phạm vi public.

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.    public:
15.        XXXX void Sketchy()
16.        {
17.            cout<<"\n A's Sketchy()";
18.            Sketchy(-1);
19.        }
20.        YYYY void Sketchy(int num)
21.        {
22.            cout<<"\n A's Sketchy("<<num<<")";
23.        }
24.};
```

Phương thức `Sketchy(int)` được khai báo trong phạm vi `public`.

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.    public:
15.        XXXX void Sketchy()
16.        {
17.            cout<<"\n A's Sketchy()";
18.            Sketchy(-1);
19.        }
20.        YYYY void Sketchy(int num)
21.        {
22.            cout<<"\n A's Sketchy("<<num<<")";
23.        }
24.};
```

Trong thân phương thức Sketchy() có lời gọi thực hiện phương thức Sketchy(int) với đối số đầu vào là -1.

Ví dụ dẫn nhập

```
25.class B:public A
26.{
27.    public:
28.        void Sketchy()
29.        {
30.            cout<<"\n B's Sketchy()";
31.            Sketchy(-2);
32.        }
33.        void Sketchy(int num)
34.        {
35.            cout<<"\n B's Sketchy("<<num<<")";
36.        }
37.};
```

Dòng số 25 được đọc là: lớp đối tượng B kế thừa từ lớp đối tượng A với từ khóa dẫn xuất public.

Ví dụ dẫn nhập

```
25.class B:public A
26.{
27.    public:
28.        void Sketchy()
29.        {
30.            cout<<"\n B's Sketchy()";
31.            Sketchy(-2);
32.        }
33.        void Sketchy(int num)
34.        {
35.            cout<<"\n B's Sketchy("<<num<<")";
36.        }
37.};
```

Phương thức Sketchy() được khai báo trong phạm vi public.

Ví dụ dẫn nhập

```
25.class B:public A
26.{
27.    public:
28.        void Sketchy()
29.        {
30.            cout<<"\n B's Sketchy()";
31.            Sketchy(-2);
32.        }
33.        void Sketchy(int num)
34.        {
35.            cout<<"\n B's Sketchy("<<num<<")";
36.        }
37.};
```

Phương thức `Sketchy(int)` được khai báo trong phạm vi `public`.

Ví dụ dẫn nhập

```
25.class B:public A
26.{
27.    public:
28.        void Sketchy()
29.        {
30.            cout<<"\n B's Sketchy()";
31.            Sketchy(-2);
32.        }
33.        void Sketchy(int num)
34.        {
35.            cout<<"\n B's Sketchy("<<num<<")";
36.        }
37.};
```

Trong thân phương thức `Sketchy()` có lời gọi thực hiện phương thức `Sketchy(int)` với đối số đầu vào là `-2`.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.|    public:
41.|         void Sketchy(int num)
42.|          {
43.|          |   cout<<"\n C's Sketchy("<<num<<")";
44.|          }
45.};
46.void Curious(A* wacky)
47.{
48.|    wacky->Sketchy();
49.|    ((C*)wacky)->Sketchy(123);
50.}
```

Dòng số 38 được đọc là: lớp đối tượng C kế thừa từ lớp đối tượng B với từ khóa dẫn xuất public.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.|    public:
41.|        void Sketchy(int num)
42.|        {
43.|        |   cout<<"\n C's Sketchy("<<num<<")";
44.|        }
45.};
46.void Curious(A* wacky)
47.{
48.|    wacky->Sketchy();
49.|    ((C*)wacky)->Sketchy(123);
50.}
```

Phương thức Sketchy(int) được khai báo trong phạm vi public.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.|    public:
41.|         void Sketchy(int num)
42.|          {
43.|          |   cout<<"\n C's Sketchy("<<num<<")";
44.|          }
45.};
46.void Curious(A* wacky)
47.{
48.|    wacky->Sketchy();
49.|    ((C*)wacky)->Sketchy(123);
50.}
```

Dòng 46 định nghĩa hàm `Curious`.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.|    public:
41.|        void Sketchy(int num)
42.|         {
43.|         |   cout<<"\n C's Sketchy("<<num<<")";
44.|         }
45.};
46.void Curious(A* wacky)
47.{
48.|    wacky->Sketchy();
49.|    ((C*)wacky)->Sketchy(123);
50.}
```

Hàm Curious có một tham số đầu vào có tên là wacky và wacky là con trỏ đối tượng thuộc lớp A lớn.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.    public:
41.        void Sketchy(int num)
42.        {
43.           cout<<"\n C's Sketchy("<<num<<")";
44.        }
45.};
46.void Curious(A* wacky)
47.{
48.    wacky->Sketchy();
49.    ((C*)wacky)->Sketchy(123);
50.}
```

Dòng 48 đọc là: con trỏ đối tượng wacky gọi thực hiện phương thức Sketchy không có tham số.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.    public:
41.        void Sketchy(int num)
42.        {
43.           cout<<"\n C's Sketchy("<<num<<")";
44.        }
45.};
46.void Curious(A* wacky)
47.{
48.    wacky->Sketchy();
49.    ((C*)wacky)->Sketchy(123);
50.}
```

Dòng 49 đọc là: con trỏ đối tượng wacky được ép kiểu thành con trỏ đối tượng thuộc lớp C.

Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.    public:
41.        void Sketchy(int num)
42.        {
43.           cout<<"\n C's Sketchy("<<num<<")";
44.        }
45.};
46.void Curious(A* wacky)
47.{
48.    wacky->Sketchy();
49.    ((C*)wacky)->Sketchy(123);
50.}
```

Dòng 49 đọc là: con trỏ đối tượng wacky được ép kiểu thành con trỏ đối tượng thuộc lớp C.

Sau đó con trỏ đối tượng wacky gọi thực hiện phương thức Sketchy có tham số với đối số là 123.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 53 đọc là: `inky` là con trỏ đối tượng thuộc lớp đối tượng `A`.
- `new B` là tạo ra đối tượng thuộc lớp đối tượng `B`.
- Địa chỉ của đối tượng vừa được tạo ra được gán cho con trỏ đối tượng `inky`.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 54 đọc là: con trỏ đối tượng `inky` gọi thực hiện phương thức `Sketchy` không có tham số.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 55 đọc là: con trỏ đối tượng `inky` gọi thực hiện phương thức `Sketchy` có tham số với đối số là `23`.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 56 đọc là: Hàm `Curiuos` được gọi thực hiện với đối số là `inky`.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 57 đọc là: pinky là con trỏ đối tượng thuộc lớp đối tượng B.
- new C là tạo ra đối tượng thuộc lớp đối tượng C.
- Địa chỉ của đối tượng vừa được tạo ra được gán cho con trỏ đối tượng pinky.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 58 đọc là: con trỏ đối `pinky` gọi thực hiện phương thức `Sketchy` không có tham số.

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 59 đọc là: con trỏ đối tượng `pinky` gọi thực hiện phương thức `Sketchy` có tham số với đối số là `46`.

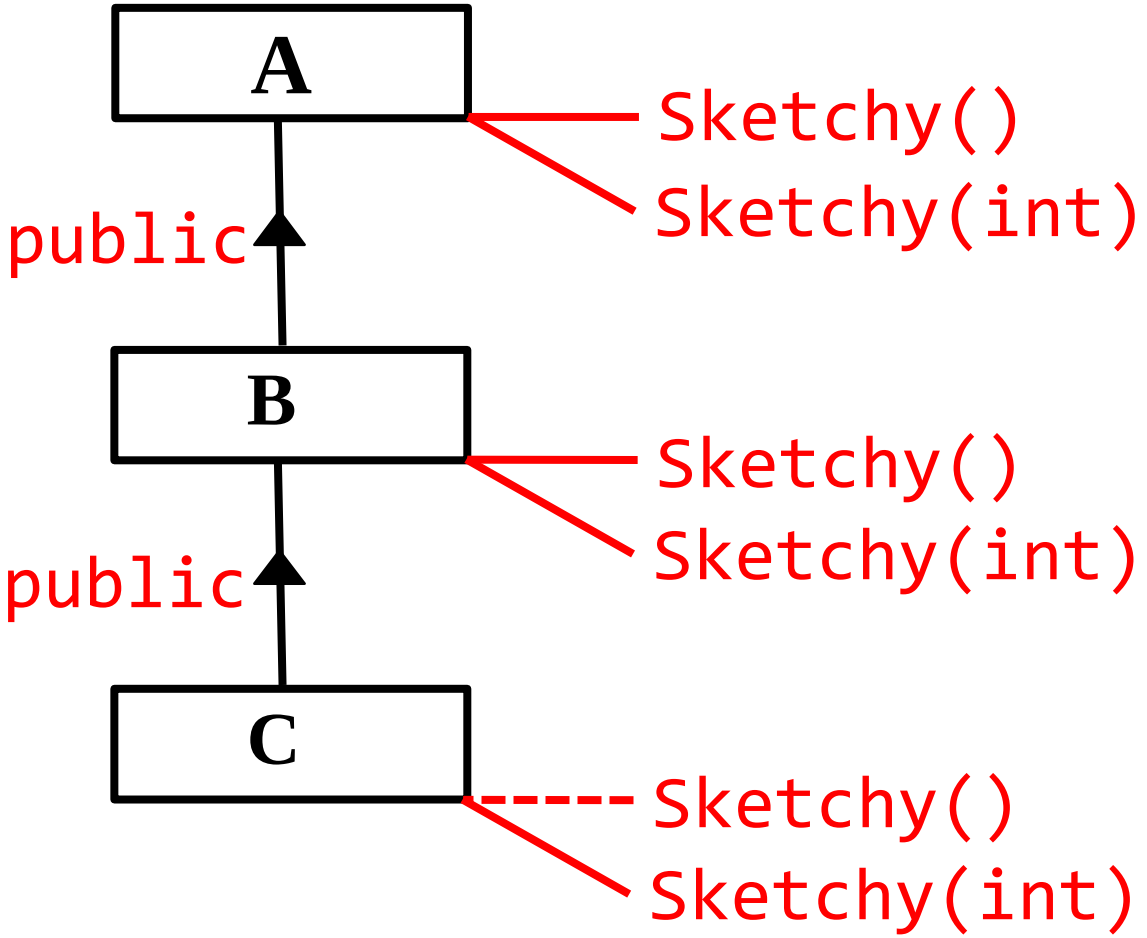
Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```

- Dòng 60 đọc là: Hàm `Curiuos` được gọi thực hiện với đối số là `pinky`.

Ví dụ dẫn nhập



Ví dụ dẫn nhập

- Hãy thực hiện chương trình dưới đây và cho biết kết quả của việc chạy chương trình trong bốn trường hợp:
- Trường hợp 1: **XXXX** là khoảng trắng,
  YYYY là khoảng trắng.
- Trường hợp 2: **XXXX** là `virtual`,
  YYYY là khoảng trắng.
- Trường hợp 3: **XXXX** là khoảng trắng,
  YYYY là `virtual`.
- Trường hợp 4: **XXXX** là `virtual`,
  YYYY là `virtual`.

Ví dụ dẫn nhập

```
11.#include <iostream>
12.class A
13.{
14.|    public:
15.|        XXXX void Sketchy()
16.|        {
17.|        |    cout<<"\n A's Sketchy()";
18.|        |    Sketchy(-1);
19.|        }
20.|        YYYY void Sketchy(int num)
21.|        {
22.|        |   cout<<"\n A's Sketchy("<<num<<")";
23.|        }
24.};
```

Ví dụ dẫn nhập

```
25.class B:public A
26.{
27.|   public:
28.|       void Sketchy()
29.|       {
30.|       |   cout<<"\n B's Sketchy()";
31.|       |   Sketchy(-2);
32.|       }
33.|       void Sketchy(int num)
34.|       {
35.|       |   cout<<"\n B's Sketchy("<<num<<")";
36.|       }
37.};
```

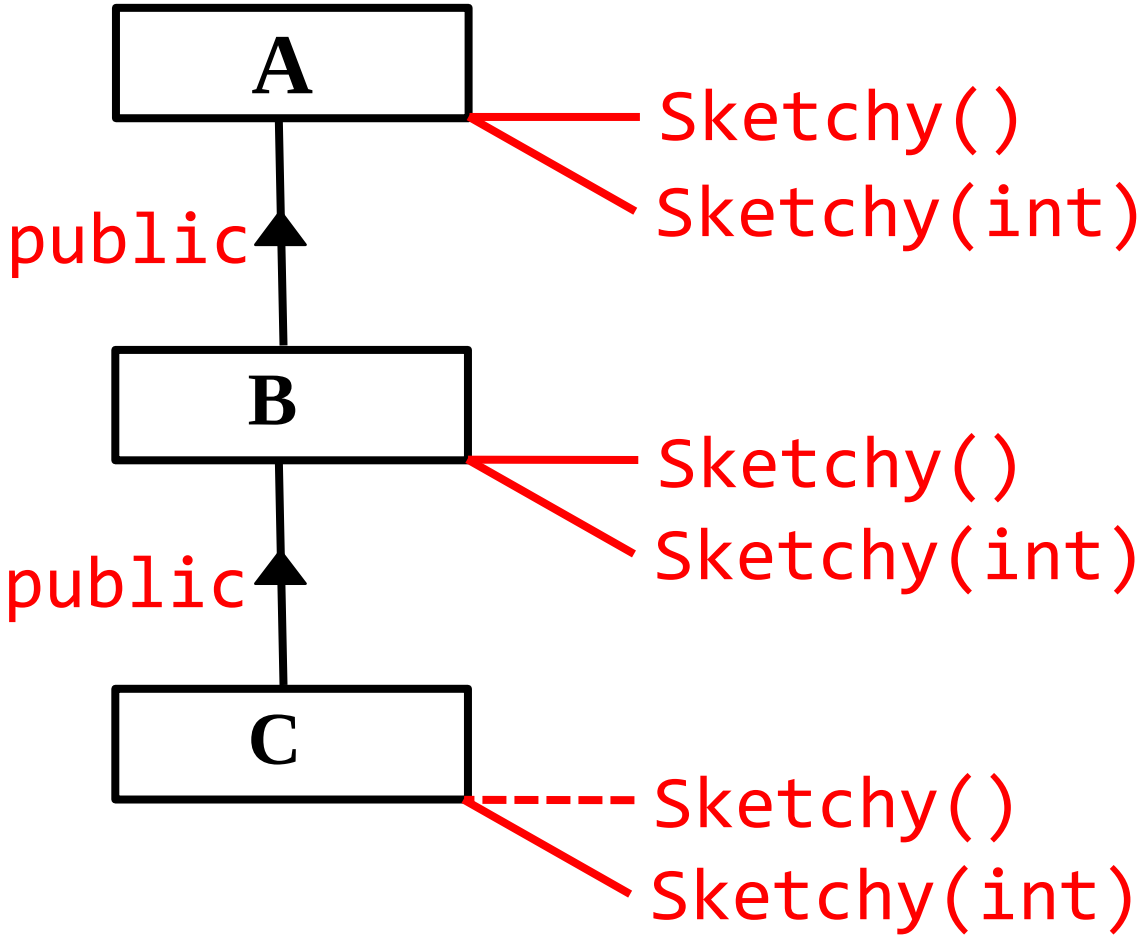
Ví dụ dẫn nhập

```
38.class C:public B
39.{
40.|    public:
41.|        void Sketchy(int num)
42.|         {
43.|         |   cout<<"\n C's Sketchy("<<num<<")";
44.|         }
45.};
46.void Curious(A* wacky)
47.{
48.|    wacky->Sketchy();
49.|    ((C*)wacky)->Sketchy(123);
50.}
```

Ví dụ dẫn nhập

```
51.void main()
52.{
53.    A* inky = new B;
54.    inky->Sketchy();
55.    inky->Sketchy(23);
56.    Curious(inky);

57.    B* pinky = new C;
58.    pinky->Sketchy();
59.    pinky->Sketchy(46);
60.    Curious(pinky);
61.}
```



3. KHÁI NIỆM ĐA XẠ

Khái niệm đa xạ

- Khái niệm: Đa xạ là cơ chế tầm vực động (dynamic scope) cho phép "*xác định*" đúng hành vi (phương thức – method) của đối tượng (object) khi yêu cầu thực hiện.

- Việc "*xác định*" được thực hiện theo nguyên tắc tự nhiên: đối tượng thuộc lớp nào sẽ gọi thực hiện phương thức của lớp đối tượng (class) đó.

Khái niệm đa xạ

- Tầm vực động (dynamic scope) là cơ chế gọi thực hiện phương thức thông qua con trỏ đối tượng.

Syntax

4. CÚ PHÁP ĐA XẠ

Cú pháp đa xạ

```
11.class CCoSo
12.{
13.    private:
14.        ...
15.    protected:
16.        ...
17.    public:
18.        ...
19.        virtual KDL <TenPhuongThuc>(<ThamSo>);
20.};
```

Cú pháp đa xạ

```
11.class CDanXuat:<Từ khóa dẫn xuất> CCoSo
12.{
13.|    private:
14.|        ...
15.|    protected:
16.|        ...
17.|    public:
18.|        ...
19.|        KDL <TenPhuongThuc>(<ThamSo>);
20.};
```

```
11. class CCoSo
12. {
13.     private: …
14.     protected: …
15.     public: …
16.         ...
17.         virtual KDL <TenPhuongThuc>(<ThamSo>);
18. };
19. class CDanXuat::<Từ khóa dẫn xuất> CCoSo
20. {
21.     private: …
22.     protected: …
23.     public: …
24.         ...
25.         KDL <TenPhuongThuc>(<ThamSo>);
26. };
```

Cú pháp đa xạ

- Một phương thức được khai báo bắt đầu với từ khóa `virtual` thì được gọi là phương thức ảo.
- Phương thức ảo được gọi thực hiện theo cơ chế đa xạ nếu lời gọi thực hiện phương thức được thông qua một con trỏ đối tượng.

- Hơn nữa, các phương thức ở lớp dẫn xuất cùng tên và cùng danh sách tham số đầu vào thì cũng sẽ là phương thức ảo nếu ở lớp cơ sở phương thức cùng tên và cùng tham số là phương thức ảo.

Khái niệm đa xạ

- Khái niệm: Đa xạ là cơ chế tầm vực động (dynamic scope) cho phép "*xác định*" đúng hành vi (phương thức – method) của đối tượng (object) khi yêu cầu thực hiện.
- Việc "*xác định*" được thực hiện theo nguyên tắc tự nhiên: đối tượng thuộc lớp nào sẽ gọi thực hiện phương thức của lớp đối tượng (class) đó.
- Tầm vực động (dynamic scope) là cơ chế gọi thực hiện phương thức thông qua con trỏ đối tượng.

5. NGUYÊN LÝ ĐA XẠ

Qui tắc 01

Qui tắc 1: Con trỏ đối tượng khi gọi thực hiện một phương thức, nếu phương thức đó là phương thức không ảo thì nó sẽ gọi thực hiện phương thức của lớp đối tượng mà con trỏ thuộc về.

Qui tắc 02

Qui tắc 2: Con trỏ đối tượng khi gọi thực hiện một phương thức, nếu phương thức đó là phương thức ảo thì chương trình sẽ xem xét xem con trỏ đối tượng đang giữ địa chỉ của đối tượng thuộc về lớp đối tượng nào và chương trình sẽ gọi thực hiện phương thức của lớp đối tượng đó.

Cảm ơn quí vị đã lắng nghe

Nhóm tác giả
TS. Nguyễn Tấn Trần Minh Khang