LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG C++

Tính thừa kế (Inheritance)

Giảng viên: ThS. Nguyễn Đỗ Thái Nguyên
E-mail: nguyenndt@hcmup.edu.vn

Nội dung

- Ví dụ mở đầu
- Khái niệm về tính thừa kế
  - Thừa kế đơn
  - Thừa kế bội
- Xây dựng lớp thừa kế
- Phương thức thiết lập và phương thức hủy bỏ

Ví dụ mở đầu

- **Xét bài toán sau: Cho một hình vuông và một hình chữ nhật. Hãy so sánh diện tích của chúng.**
- **Phân tích:**
  - **Đối tượng: hình vuông, hình chữ nhật**
  - **Lớp: HV, HCN**

Ví dụ mở đầu (tt)

- **Thiết kế:**
  - **Lớp HV:**
    - **Dữ liệu:**
      - **Độ dài cạnh: là một số thực**
    - **Phương thức:**
      - **Phương thức thiết lập, phương thức hủy bỏ**
      - **Phương thức Set: thiết lập độ dài cạnh hình vuông**
      - **Phương thức Dt: tính diện tích hình vuông**

Ví dụ mở đầu (tt)

- **Thiết kế (tt):**
  - **Lớp HCN:**
    - **Dữ liệu:**
      - **Độ dài, độ rộng: là hai số thực**
    - **Phương thức:**
      - **Phương thức thiết lập, phương thức hủy bỏ**
      - **Phương thức Set: thiết lập độ dài và độ rộng của hình chữ nhật**
      - **Phương thức Dt: tính diện tích hình chữ nhật**

Ví dụ mở đầu (tt)

HV
// Độ dài cạnh float canh;

HV(float x=0);
HV(HV &h);
~HV();
 	
void Set(float);
float DT();

HCN
// Độ dài, độ rộng float dai, rong;

HCN(float x=0, float y=0);
HCN(HCN &h);
~HCN();
 	
void Set(float, float);
float DT();

Ví dụ mở đầu (tt)

```
class HV
{
      float canh;
   public:
      HV (float x = 0)
      {
          printf("Hinh vuong duoc tao \n");
          canh = x;
      }
      HV(HV &h)
      {
          printf("Hinh vuong duoc copy \n");
          canh = h.canh;
      }
      void Set(float x)
      {    canh = x;  }
      float Dt()
      {
           return canh*canh;
      }
};
```

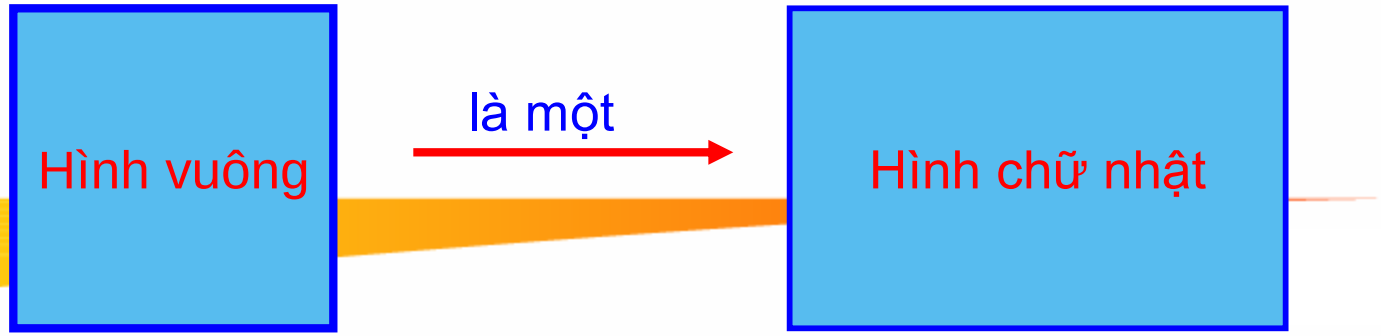
```
class HCN
{
      float dai, rong;
   public:
      HV (float x = 0, float y = 0)
      {
          printf("Hinh CN duoc tao \n");
          dai = x;     rong = y;
      }
      HCN(HCN &h)
      {
          printf("Hinh CN duoc copy \n");
          dai = h.dai;     rong = h.rong;
      }
      void Set(float x, float y)
      {    dai = x;  rong = y}
      float Dt()
      {
           return dai*rong;
      }
};
```

Ví dụ mở đầu (tt)

```
void main()
{
    HV hv;
    HCN hcn;
    float x, y;
    printf(“Hay nhap do dai canh hinh vuong: ”);     scanf(“%f”,&x);
    hv.Set(x);
    printf(“Hay nhap do dai, do rong cua hinh chu nhat: ”);     scanf(“%f %f”,&x,&y);
    hcn.Set(x,y);
    if (hv.DT() > hcn.DT())
        printf(“ DT hinh vuong lon hon ”);
    else
        if (hv.DT() < hcn.DT())
            printf(“ DT hinh chu nhat lon hon ”);
        else
            printf(“ DT hai hinh bang nhau ”);
   getch();
}
```

Ví dụ mở đầu (tt)

- **Nhận xét:**
  - **Các hàm bị lặp lại giữa các lớp.**
  - **Làm thế nào để tận dụng khái niệm đã biết: "hình vuông là một hình chữ nhật có độ dài và độ rộng bằng nhau"???**
  - **Giải pháp: sử dụng kế thừa để lớp HV có thể tận dụng những dữ liệu và phương thức của hình chữ nhật**



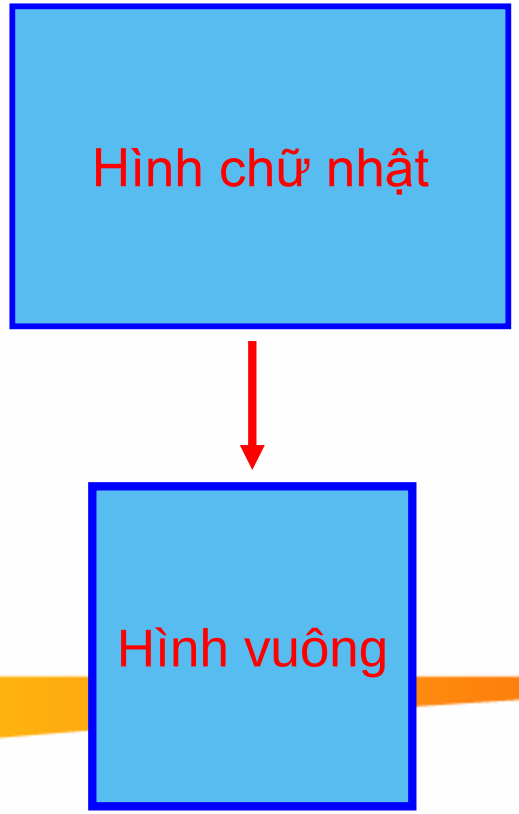
Tính thừa kế? (Inheritance)

Khái niệm về tính thừa kế x

- Tính thừa kế là một kỹ thuật cho phép một lớp có thể thừa hưởng dữ liệu và phương thức của một lớp khác
- Nếu lớp A được thừa kế từ lớp B thì ta nói A là lớp con (child class, subclass) và B là lớp cha (parent class, superclass)
- Một đối tượng thuộc lớp con, nếu được phép, sẽ có quyền sử dụng dữ liệu và phương thức của lớp cha mà không cần phải định nghĩa lại

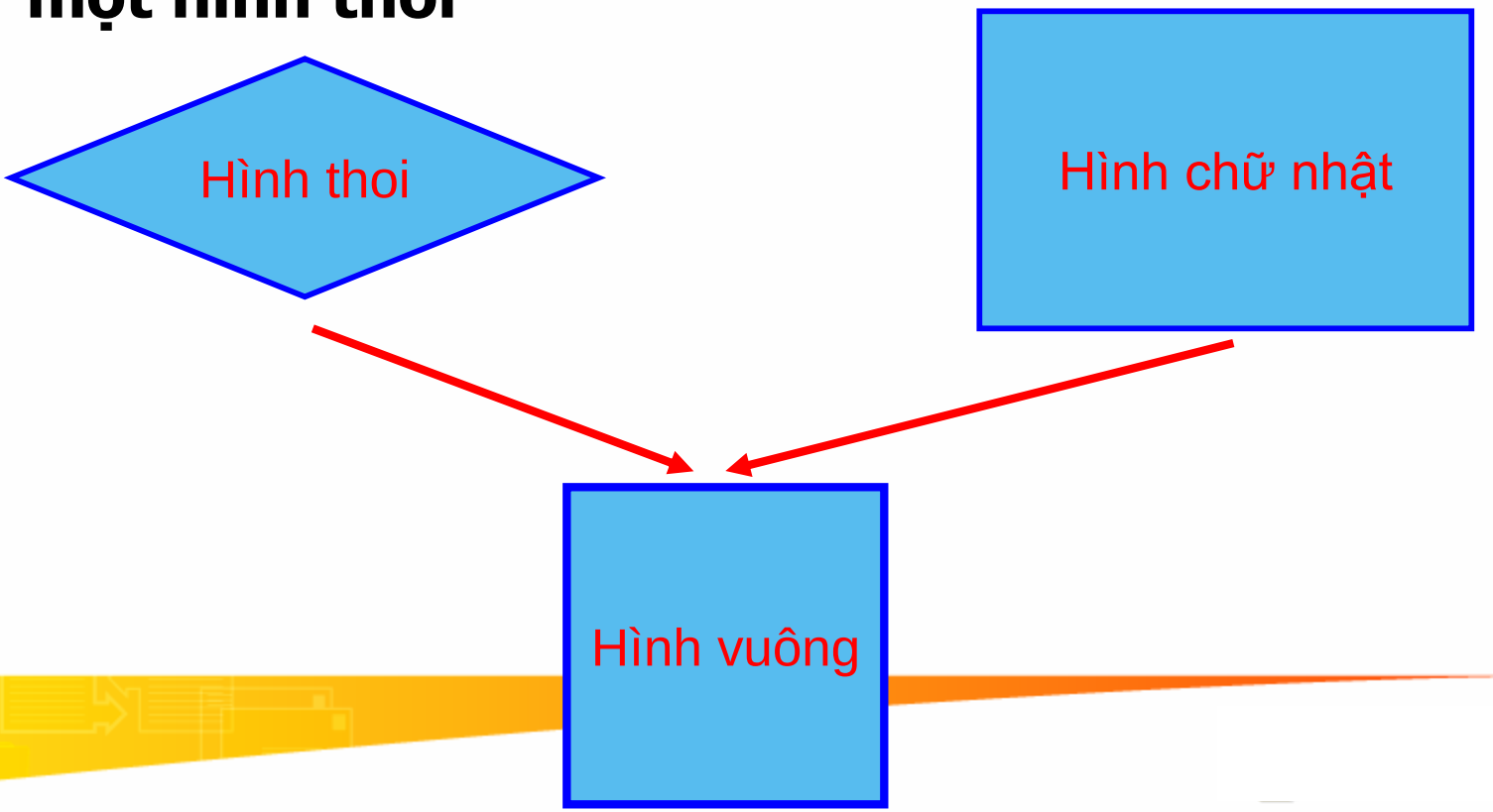
Khái niệm về tính thừa kế

- **Thừa kế đơn: một lớp con thừa kế từ một lớp cha.**
  - **VD: Hình vuông là một hình chữ nhật**



Khái niệm về tính thừa kế

- **Thừa kế bội: một lớp con thừa kế từ hai lớp cha trở lên.**
  - **VD: Hình vuông là một hình chữ nhật và cũng là một hình thoi**



Khái niệm về tính thừa kế x

- **Thuộc tính thừa kế: private, protected, public**
- **Bảng giao thuộc tính thừa kế và thuộc tính trên dữ liệu:**

TT thừa kế / TT dữ liệu	private	protected	public
private	private	private	private
protected	private	protected	protected
public	private	protected	public

Xây dựng lớp thừa kế

- **Xây dựng lớp thừa kế cũng bao gồm các thiết kế giống như một lớp bình thường**
- **Khi muốn thừa kế lớp nào thì khai báo thêm bằng cách dùng dấu hai chấm (:) ngay bên cạnh tên lớp mới khai báo. Cụ thể:**

```
class ChildClass : <TT thừa kế> ParentClass
{
        ...
};
```

Xây dựng lớp thừa kế đơn

```
class HCN
{
      float dai, rong;
   public:
      HCN (float x = 0, float y = 0)
      {
          printf("Hinh CN duoc tao \n");
          dai = x;     rong = y;
      }
      HCN(HCN &h)
      {
          printf("Hinh CN duoc copy \n");
          dai = h.dai;     rong = h.rong;
      }
      void Set(float x, float y)
      {    dai = x;  rong = y}
      float Dt()
      {
           return dai*rong;
      }
};
```

```
class HV : public HCN
{
      float canh;
   public:
      HV (float x = 0)
      {
          printf("Hinh vuong duoc tao \n");
          canh = x;  HCN::Set(x,x);
      }
      HV(HV &h)
      {
          printf("Hinh vuong duoc copy \n");
          canh = h.canh; HCN::Set(canh,canh);
      }
      void Set(float x)
      {    canh = x;  HCN::Set(x,x);        }
      // float Dt()
      // {
      //       return canh*canh;
      // }
};
```

Xây dựng lớp thừa kế đơn(tt)

```
void main()
{
    HV hv;
    HCN hcn;
    float x, y;
    printf("Hay nhap do dai canh hinh vuong: ");     scanf("%f",&x);
    hv.Set(x);
    printf("Hay nhap do dai, do rong cua hinh chu nhat: ");     scanf("%f %f",&x,&y);
    hcn.Set(x,y);
    if (hv.DT() > hcn.DT())
        printf(" DT hinh vuong lon hon ");
    else
        if (hv.DT() < hcn.DT())
            printf(" DT hinh chu nhat lon hon ");
        else
            printf(" DT hai hinh bang nhau ");
   getch();
}
```

Xây dựng lớp thừa kế bội

- Giống thừa kế đơn, chỉ cần liệt kê thêm DS các lớp cha để thừa kế
- Các đặc điểm của thừa kế đơn vẫn đúng cho thừa kế bội.
- Cần lưu ý khi có hai thành phần (dữ liệu hoặc phương thức) giống nhau giữa hai lớp cha thì cần rạch ròi thành phần nào được sử dụng bằng phép sử dụng toán tử :: x

Xây dựng lớp thừa kế bội (tt)

```
class HinhThoi
{
      float canh, goc;
   public:
      HinhThoi (float x = 0, float y = 0)
      {
          printf("Hinh thoi duoc tao \n");
          canh = x;     goc = y;
      }
      HinhThoi(HinhThoi &h)
      {
          printf("Hinh thoi duoc copy \n");
          canh = h.canh;   goc = h.goc;
      }
      void Set(float x, float y)
      {    canh = x;  goc = y}
      float Dt()
      {
            return canh*canh*sin(goc*Pi/180);
      }
};
```

```
class HV : public HCN, public HinhThoi
{
      float canh;
   public:
      HV (float x = 0)
      {   printf("Hinh vuong duoc tao \n");
          canh = x;  HCN::Set(x,x);
          HinhThoi::Set(x,90);
      }
      HV(HV &h)
      {    printf("Hinh vuong duoc copy \n");
          canh = h.canh; HCN::Set(canh,canh);
          HinhThoi::Set(canh,90);
      }
      void Set(float x)
      {    canh = x;  HCN::Set(x,x);
           HinhThoi::Set(x,90);    }
      // float Dt()
      // {
      //       return canh*canh;
      // }
};
```

Xây dựng lớp thừa kế đơn(tt)

```
void main()
{
    HV hv;
    HCN hcn;
    float x, y;
    printf("Hay nhap do dai canh hinh vuong: ");    scanf("%f",&x);
    hv.Set(x);
    printf("Hay nhap do dai, do rong cua hinh chu nhat: ");    scanf("%f %f",&x,&y);
    hcn.Set(x,y);
    if (hv.DT() > hcn.DT())
        printf(" DT hinh vuong lon hon ");
    else
        if (hv.DT() < hcn.DT())
            printf(" DT hinh chu nhat lon hon ");
        else
            printf(" DT hai hinh bang nhau ");
   getch();
}
```

Báo lỗi do không biết phải gọi phương thức tính diện tích của HCN hay HinhThoi

Phương thức thiết lập và phương thức hủy bỏ

- Khi một đối tượng được sinh ra, không những phương thức thiết lập của nó được gọi tự động mà phương thức thiết lập của cha nó cũng được gọi tự động
- Tương tự như vậy, khi đối tượng bị hủy, phương thức hủy bỏ của cha nó cũng tự động được gọi
- Lớp con không được truy cập (gọi) đến phương thức thiết lập và phương thức hủy bỏ của lớp cha

Phương thức thiết lập và phương thức hủy bỏ (tt) x

- **Thứ tự gọi phương thức thiết lập: cha trước – con sau.**
- **Thứ tự gọi phương thức hủy bỏ: con trước – cha sau.**

Phương thức thiết lập và phương thức hủy bỏ (tt)

```
class LopA
{   public:
        LopA ()
        {
            printf("A duoc tao \n");
        }
        ~LopA()
        {
             printf("A bi huy \n");
        }
};
class LopB : public LopA
{  public:
        LopB ()
        {
            printf("B duoc tao \n");
        }
        ~LopB()
        {
             printf("B bi huy \n");
        }
};
```

```
class LopC : public LopB
{  public:
        LopC ()
        {
            printf("C duoc tao \n");
        }
        ~LopC()
        {
             printf("C bi huy \n");
        }
};

void main()
{
    {
        LopC c;
    }
    getch();
}
```

Dự đoán kết quả hiện lên trên màn hình

Phương thức thiết lập và phương thức hủy bỏ (tt)

```
class LopA
{   public:
        LopA ()
        {
            printf("A duoc tao \n");
        }
        ~LopA()
        {
             printf("A bi huy \n");
        }
};
class LopB : public LopA
{  public:
        LopB ()
        {
            printf("B duoc tao \n");
        }
        ~LopB()
        {
             printf("B bi huy \n");
        }
};
```

```
class LopC : public LopA
{  public:
        LopC ()
        {
            printf("C duoc tao \n");
        }
        ~LopC()
        {
             printf("C bi huy \n");
        }
};

void main()
{
    {
        LopC c;
    }
    getch();
}
```

Dự đoán kết quả hiện lên trên màn hình

Bài tập áp dụng

- **Xây dựng lớp TAMGIAC và lớp TAMGIACDEU được kế thừa từ lớp TAMGIAC với đặc điểm cộng thêm là cả 3 cạnh đều bằng nhau**

Bài tập áp dụng (tt)

TAMGIAC

```
// Độ dài cạnh
float a,b,c;
TAMGIAC(float,float,float)
;
TAMGIAC(TAMGIAC &h);
~TAMGIAC();

void Set(float,float,float);
float DT();
```

TG_DEU

```
// Độ dài, độ rộng
float canh;
TG_DEU(float);
TG_DEU(TG_DEU &h);
~TG_DEU();

void Set(float);
float DT();
```

Câu hỏi và thảo luận

Chân thành cảm ơn !