Hàm thuần, lớp trừu tượng

Hàm thuần: một khái niệm của LT hướng đối tượng, dùng để mô tả một thành phần của lớp trong thừa kế. Một hàm thuần thì không có định nghĩa nội dung.

Khai báo một hàm thuần: void <pure_func>() = 0;

Vì khai báo là hàm thuần, nó không thể thực hiện theo cách thông thường. Hàm thuần không sử dụng trực tiếp, và để mô tả một thành phần của lớp trong thừa kế.

Lớp trừu tượng: có chứa ít nhất một hàm thuần, nó không được sử dụng như cách thông thường, chỉ để khai báo hình thức. Trong các ngôn ngữ OOP như Visual Basic, lớp trừu tượng mô tả như tầng lớp interface.

Khai báo một lớp trừu tượng:

```
class <abstract_cls>
{
  private:
  //..
  public:
  void <pure_func>() = 0;
};
```

Ví dụ: Mô tả lớp trừu tượng Đa giác, có khai báo hàm thuần tính diện tích đa giác.

```
class Polygon
{
   int num;
   float  *vertices;
   public:
   float Area() = 0;
};
```

Khai báo đối tượng lớp trừu tượng: Biến đối tượng là biến trỏ địa chỉ, không sử dụng trực tiếp với đối tượng cơ sở.

Để thao tác với biến trỏ đối tượng lớp trừu tượng, phải khai báo lớp dẫn xuất từ lớp trừu tượng. Bên trong lớp dẫn xuất có định nghĩa lại cụ thể hàm thành viên cho hàm thuần lớp cơ sở, và lớp dẫn xuất không khai báo thêm hàm thuần khác.

Ví dụ: Khai báo lớp dẫn xuất

```
class Triangle : public Polygon
{
   float p,a,b,c;  // xác định độ dài cạnh và chu vi
                   // từ số đỉnh và tọa độ từ dữ liệu ban đầu num, vertices
   public:
   float Area()  // Định nghĩa lại chức năng
```

```
  {
    p = (a+b+c)/2;
    return sqrt(p*(p-a)*(p-b)*(p-c));
  }
};
```

Khai báo đối tượng:
Polygon plg; //error
Polygon *ptr; //ok
cout<<"Dien tich da giac: "<<ptr->Area()<<endl; //error
Triangle triangle; //ok
ptr = ▵ // ok

Hàm ảo, đa hình
Một lớp cơ sở là lớp đa giác, dẫn xuất từ nó là các lớp tứ giác, hình tròn. Một **đối tượng lớp cơ sở thích nghi với các đối tượng của những lớp dẫn xuất**.
Cho khai báo:
class Polygon
{};
class Cỉrcle: public Polygon
{}:
class Rectangle: public Polygon
{}:
Polygon plg;
Circle c;
Rectangle rect;
plg = c; //ok
plg = rect; //ok

Hạn chế: Đối tượng plg chỉ có thể chứa dữ liệu của lớp cơ sở và thực hiện chức năng lớp cơ sở, không truy nhập được chức năng riêng của đối tượng dẫn xuất. Cụ thể, một đối tượng lớp Polygon không xác định được thông tin riêng (chu vi hoặc diện tích) của một đối tượng lớp Circle.
OOP cung cấp khái niệm hàm ảo.
Nếu như lớp cơ sở khai báo một **hàm ảo** thành viên và lớp dẫn xuất **định nghĩa lại chức năng** cho hàm thành viên của lớp cơ sở, thì đối tượng lớp cơ sở sẽ truy nhập và thực hiện chức năng riêng của những đối tượng dẫn xuất.

Ví dụ:

```
class Polygon
{
   int num;
   float  *vertices;
   public:
   virtual float Area() = 0;  //hàm ảo thuần
};
class Triangle : public Polygon
{
   float p,a,b,c;  // xác định độ dài cạnh và chu vi
                   // từ số đỉnh và tọa độ từ dữ liệu ban đầu num, vertices
   public:
   float Area()  // Định nghĩa lại chức năng, không bắt buộc có virtual
   {
      p = (a+b+c)/2;
      return sqrt(p*(p-a)*(p-b)*(p-c));
   }
};

int main()
{
   Polygon *ptr;
   Triangle triangle;
   ptr = &triangle;
   cout<<"Dien tich Triamgle: "<<ptr->Area()<<endl; //ok
}
```

Cơ chế bảng ảo lưu trữ địa chỉ của các hàm thành viên Area(). Khi đó, biến trỏ đối tượng cơ sỡ truy nhập địa chỉ mỗi chức năng thành viên của đối tượng dẫn xuất (dựa theo địa chi bảng ảo đã lưu trữ). Do vậy, cơ chế trên gọi là đa hình hay ràng buộc động.

Ví dụ: Mục 5.6.2 Bài giảng