Bài 14: OOP

# Khái niệm

OOP, hay Object-Oriented Programming (Lập trình hướng đối tượng), là một phương pháp lập trình mà trong đó mọi thứ được tổ chức xung quanh các "đối tượng". Mỗi đối tượng là một thể hiện của một lớp class, và class định nghĩa cách mà các đối tượng sẽ hoạt động.

# Bốn đặc tính của OOP

1. Tính đóng gói ( Encapsulation):

Tính đóng gói ( Encapsulation) là ẩn đi các property “ mật” khỏi người dùng. Và để làm được điều này, ta sẽ khai báo các property ở quyền truy cập private ( tức là không thể truy cập trực tiếp tới các property này).

Trong trường hợp ta muốn đọc hoặc ghi các property này, thì ta cung cấp các method ở quyền truy cập public.

Ví dụ:

| #**include** <iostream> #**include** <string>  **using** **namespace** std;   **class** **Student** {  **private**:  string Name;  **double** GPA;  **int** StudentID;  **public**:  Student(string name);   string **getName**()  {  **return** Name;  }   **void** **setGPA**(**double** gpa)  {  GPA = gpa;  }  **double** **getGPA**()  {  **return** GPA;  }   **int** **getID**()  {  **return** StudentID;  }  };  Student::Student(string name) {  Name = name;  **static** **int** id = 1000;  StudentID = id;  ++id; }    **int** **main**() {  Student **student1**("Trung");  Student **student2**("Thai");  Student **student3**("Thao");   cout << "ID: " << student1.getID() << endl;  cout << "Name: " << student1.getName() << endl;   cout << "ID: " << student2.getID() << endl;  cout << "Name: " << student2.getName() << endl;   cout << "ID: " << student3.getID() << endl;  cout << "Name: " << student3.getName() << endl;     **return** 0; } |
| --- |

1. Tính kế thừa

Tính kế thừa ( Inheritance) là khả năng sử dụng lại các property và method của một class trong một class khác. Ta chia chúng làm 2 loại là class cha và class con. Để kế thừa từ class khác, ta dùng ký tự “ : ”.

Tất cả những property và method có quyền truy cập là public và protected ở class cha sẽ được class con kế thừa.

Có 3 kiểu kế thừa là public, private và protected. Những property và method được kế thừa từ class cha sẽ nằm ở quyền truy cập của class con tương ứng với kiểu kế thừa.

Ví dụ:

| #**include** <iostream> #**include** <string>   **using** **namespace** std;  **class** **Person** { **protected**:  string Name;  **int** Age;  string Home\_Address;  **public**:  string **getName**()  {  **return** Name;  }  **void** **setName**(string name)  {  Name = name;  }   **int** **getAge**()  {  **return** Age;  }  **void** **setAge**(**int** age)  {  Age = age;  }   string **getAddress**()  {  **return** Home\_Address;  }  **void** **setAddress**(string address)  {  Home\_Address = address;  }   **void** **displayInfo**()  {  cout << "Name: " << Name << endl;  cout << "Age: " << Age << endl;  cout << "Address: " << Home\_Address << endl;  }   };   **class** **Student** : **public** Person { **private**:  string School\_Name;  **double** GPA;  **int** StudentID;  **public**:  Student()  {  **static** **int** id = 1000;  StudentID = id;  id++;  }   string **getSchoolName**()  {  **return** School\_Name;  }  **void** **setSchoolName**(string school\_name)  {  School\_Name = school\_name;  }   **double** **getGPA**()  {  **return** GPA;  }  **void** **setGPA**(**double** gpa)  {  GPA = gpa;  }   **int** **getID**()  {  **return** StudentID;  } };   **int** **main**() {  Person person1;  person1.setName("Trung");  person1.setAge(20);  person1.setAddress("HCM");  person1.displayInfo();   cout << "-----------------------" << endl;    Student student1;   student1.setName("Trungg");  student1.setAge(24);  student1.setAddress("HCMM");  student1.setGPA(8.1);  student1.setSchoolName("DinhTienHoang");  student1.displayInfo();   **return** 0; } |
| --- |

Ở ví dụ trên, class Student đã kế thừa các method của class Person. Tuy nhiên, ở method displayInfo(), thì chỉ in ra 3 thông tin là Name, Age và Address. Để method displayInfo() ở class Student có thể in thêm thông tin GPA và School, ta dùng kỹ thuật ghi đè ( overriding) như sau:

| #**include** <iostream> #**include** <string>   **using** **namespace** std;  **class** **Person** { **protected**:  string Name;  **int** Age;  string Home\_Address;  **public**:  string **getName**()  {  **return** Name;  }  **void** **setName**(string name)  {  Name = name;  }   **int** **getAge**()  {  **return** Age;  }  **void** **setAge**(**int** age)  {  Age = age;  }   string **getAddress**()  {  **return** Home\_Address;  }  **void** **setAddress**(string address)  {  Home\_Address = address;  }   **void** **displayInfo**()  {  cout << "Name: " << Name << endl;  cout << "Age: " << Age << endl;  cout << "Address: " << Home\_Address << endl;  }   };   **class** **Student** : **public** Person { **private**:  string School\_Name;  **double** GPA;  **int** StudentID;  **public**:  Student()  {  **static** **int** id = 1000;  StudentID = id;  id++;  }   string **getSchoolName**()  {  **return** School\_Name;  }  **void** **setSchoolName**(string school\_name)  {  School\_Name = school\_name;  }   **double** **getGPA**()  {  **return** GPA;  }  **void** **setGPA**(**double** gpa)  {  GPA = gpa;  }   **int** **getID**()  {  **return** StudentID;  }   **void** **displayInfo**() // overriding  {  cout << "Name: " << Name << endl;  cout << "Age: " << Age << endl;  cout << "Address: " << Home\_Address << endl;  cout << "School name: " << School\_Name << endl;  cout << "GPA: " << GPA << endl;  }  };   **int** **main**() {  Person person1;  person1.setName("Trung");  person1.setAge(20);  person1.setAddress("HCM");  person1.displayInfo();   cout << "-----------------------" << endl;    Student student1;   student1.setName("Trungg");  student1.setAge(24);  student1.setAddress("HCMM");  student1.setGPA(8.1);  student1.setSchoolName("DinhTienHoang");  student1.displayInfo();   **return** 0; } |
| --- |