Đề 1

1. Kế thừa là một tính năng quan trọng trong lập trình hướng đối tượng, cho phép lớp con (subclass) sử dụng lại các thuộc tính và phương thức của lớp cha (superclass), đồng thời có thể chỉnh sửa hoặc mở rộng chúng để đáp ứng các yêu cầu cụ thể của lớp con.

Vai trò của kế thừa là giúp tối ưu hóa việc lập trình, giảm thiểu độ phức tạp của mã nguồn, đồng thời cũng giúp tăng tính linh hoạt và tái sử dụng của mã nguồn.

Ví dụ về kế thừa trong thực tế là lớp con "Sinh viên" có thể kế thừa các thuộc tính và phương thức từ lớp cha "Người" như tên, địa chỉ, tuổi, ngày sinh, số điện thoại, email. Tuy nhiên, lớp con "Sinh viên" có thể mở rộng các thuộc tính và phương thức đặc trưng của nó như điểm trung bình, môn học, lớp học, giảng viên đứng lớp,... Bằng cách này, lớp con "Sinh viên" có thể sử dụng lại các thuộc tính và phương thức của lớp cha "

1. Phân biệt public, protected và Private:
   * Public, protected và private đề cập đến khả năng truy cập của các phương thức.
   * Mặc định, tất cả phương thức đều ở trạng thái public. Nếu không chỉ định khả năng truy cập của phương thức, nó sẽ là public.
   * Phương thức protected và private không thể truy cập một cách tự do, và do đó khi có một thể hiện của đối tượng, bạn sẽ không thể gọi được các phương thức đó.
   * Public (Công khai): đây là thuộc tính công khai, khi được khai báo vào trong biến hoặc hàm. Các thuộc tính và phương thức này có thể truy xuất ra bên ngoài class
   * Protected (Được bảo vệ): các thuộc tính các phương thức không thể truy xuất ra bên ngoài class. Nó chỉ được gọi trong class và class kế thừa.
   * Private (Riêng tư): các thuộc tính các phương thức không thể truy xuất ra bên ngoài class. Nó chỉ được gọi trong phạm vi class. (Đây là class riêng tư, chỉ dùng trong nội bộ của class đó).
   * Ví dụ:
     + Ví dụ: `class Person { public: string name; void printName() { cout << "My name is " << name << endl; }};```

Trong ví dụ này, biến name và phương thức printName đều được khai báo là public, nên các đối tượng khác có thể truy cập vào chúng.

* + - Ví dụ:```class Shape { protected: int width; int height;}; class Rectangle: public Shape { public: int getArea() { return (width \* height); }};```

Trong ví dụ này, biến width và height được khai báo là protected trong class Shape, nên class Rectangle kế thừa từ Shape có thể truy cập được vào các biến này.