ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

BÁO CÁO BÀI TẬP REVIEW
LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Nhóm sinh viên thực hiện:

Cáp Kim Hải Anh - 23520036

Nguyễn Thiên Bảo - 23520127

Đặng Quốc Cường - 23520192

Lớp: IT002.P11.CTTN.1

GVHD: Nguyễn Ngọc Quí

Ngày 10 tháng 12 năm 2024

Mục lục

1 Đề 1:

1.1 Câu 1:

1.1.1 Đề bài:

a. Phân biệt các phạm vi truy cập private, protected, public và cho ví dụ minh họa.

b. Nêu khái niệm về sự kế thừa và những ưu điểm của kế thừa trong việc lập trình. Cho ví dụ minh họa.

1.1.2 Lời giải:

a. Phân biệt các phạm vi truy cập private, protected, public và cho ví dụ minh họa:

- Thuộc tính `public`: Thành phần nào có thuộc tính `public` thì có thể truy xuất từ bất cứ nơi nào.

```
// Vi du
class MyClass {
public:
    int publicVar;
};

int main() {
    MyClass obj;
    obj.publicVar = 10; // Truy cap duoc tu ben ngoai lop
    return 0;
}
```

- Thuộc tính `private`: Thành phần có thuộc tính `private`:
  - Là riêng tư của lớp đó.
  - Chỉ có hàm thành phần của lớp và ngoại lệ các hàm bạn được phép truy xuất.
  - Các lớp con cũng không có quyền truy xuất.

```
//Vi du
class MyClass {
private:
    int privateVar;
public:
    void setVar(int val) {
        privateVar = val; // Truy cap duoc tu ben trong lop
    }
};

int main() {
    MyClass obj;
    // obj.privateVar = 10; // Loi: Khong the truy cap privateVar tu ben ngoai
    obj.setVar(10); // Su dung phuong thuc public de thay doi gia tri
    return 0;
}
```

- Thuộc tính `protected`: Cho phép quy định một vài thành phần nào đó của lớp là bảo mật, theo nghĩa thế giới bên ngoài không được phép truy xuất, nhưng tất cả các lớp con, cháu,... đều được phép truy xuất.

```cpp
// Vi du
class Base {
protected:
    int protectedVar;
};

class Derived : public Base {
public:
    void setVar(int val) {
        protectedVar = val; // Truy cap duoc tu lop dan xuat
    }
};

int main() {
    Derived obj;
    // obj.protectedVar = 10; // Loi: Khong the truy cap tu ben ngoai
    obj.setVar(10); // Su dung phuong thuc public cua lop dan xuat
    return 0;
}
```

b. Khái niệm về sự kế thừa và những ưu điểm của kế thừa trong việc lập trình. Cho ví dụ minh họa.

- **Khái niệm:** Kế thừa là một đặc điểm của ngôn ngữ dùng để biểu diễn mối quan hệ đặc biệt hóa – tổng quát hóa giữa các lớp. Các lớp được trừu tượng hóa và được tổ chức thành một sơ đồ phân cấp lớp.
- **Lợi ích của kế thừa trong việc lập trình:**
  - Kế thừa cho phép xây dựng lớp mới từ lớp đã có.
  - Kế thừa cho phép tổ chức các lớp chia sẻ mã chương trình chung, nhờ vậy có thể dễ dàng sửa chữa, nâng cấp hệ thống.
  - Trong C++, kế thừa còn định nghĩa sự tương thích, nhờ đó ta có cơ chế chuyển kiểu tự động.
- **Ví dụ minh hoạ:** Giả sử chúng ta có một lớp cơ sở Animal và lớp dẫn xuất Dog kế thừa từ Animal:

```cpp
#include <iostream>
using namespace std;

// Lop co so
class Animal {
public:
    void eat() {
        cout << "Animal is eating." << endl;
    }
};

// Lop dan xuat Dog ke thua tu Animal
class Dog : public Animal {
public:
    void bark() {
        cout << "Dog is barking." << endl;
    }
};

int main() {
    Dog myDog;
```

```
22
23      // Doi tuong Dog co the su dung phuong thuc cua Animal
24      myDog.eat();   // Output: Animal is eating.
25      myDog.bark();  // Output: Dog is barking.
26      return 0;
27  }
```

Giải thích:

- Kế thừa cho phép lớp `Dog` sử dụng lại phương thức `eat()` của lớp `Animal` mà không cần phải định nghĩa lại.
- Điều này thể hiện lợi ích của việc tái sử dụng mã (code reuse) và dễ dàng mở rộng hệ thống.
- Nếu sau này chúng ta thêm phương thức mới vào `Animal`, như `sleep()`, các lớp con như `Dog` cũng tự động kế thừa phương thức này.

1.2 Câu 2:

1.2.1 Đề bài:

a. Xét đoạn chương trình sau:

```
#include <iostream>
using namespace std;
class A {
public:
    A() {
        cout << "Constructing A ";
    }
    ~A() {
        cout << "Destructing A ";
    }
};

class B: public A {
public:
    B() {
        cout << "Constructing B ";
    }
    ~B() {
        cout << "Destructing B ";
    }
};

int main() {
    B b1;
    return 0;
}
```

Hãy cho biết kết quả xuất ra màn hình khi thực thi đoạn chương trình trên. Giải thích ngắn gọn tại sao có kết quả đó.

b. Xét đoạn chương trình sau:

```
#include <iostream>
using namespace std;
class A {
private:
    int x;
```

```
public:
    A(int t) {
        x = t;
    }
    static void f() {
        cout << x;
    }
    int f2() {
        return x;
    }
};

void main() {
    A a;
    f2(a);
}
```

Cho biết đoạn chương trình trên khi biên dịch có lỗi xảy ra hay không? Nếu có lỗi, hãy chỉ ra các lỗi đó và sửa lỗi để chương trình có thể thực thi được.

1.2.2 Lời giải:

a. Cho biết kết quả xuất ra màn hình khi thực thi đoạn chương trình trên. Giải thích ngắn gọn tại sao có kết quả đó:

- **Kết quả:** `Constructing A Constructing B Destructing B Destructing A`
- **Lý do có kết quả này là bởi:**
  - **Thứ tự gọi hàm khởi tạo (Constructor):**
    - Khi tạo một đối tượng của lớp dẫn xuất (`B`), trình biên dịch sẽ gọi constructor của lớp cơ sở (`A`) trước, sau đó mới gọi constructor của lớp dẫn xuất (`B`).
    - Điều này đảm bảo rằng tất cả các thành phần của lớp cơ sở được khởi tạo trước khi sử dụng trong lớp dẫn xuất.
  - **Thứ tự gọi hàm hủy (Destructor):**
    - Khi đối tượng bị hủy, destructor của lớp dẫn xuất (`B`) được gọi trước, sau đó mới đến destructor của lớp cơ sở (`A`).
    - Điều này đảm bảo rằng các tài nguyên được cấp phát trong lớp dẫn xuất được giải phóng trước khi lớp cơ sở bị hủy.

b. Cho biết đoạn chương trình trên khi biên dịch có lỗi xảy ra hay không? Nếu có lỗi, hãy chỉ ra các lỗi đó và sửa lỗi để chương trình có thể thực thi được:

- **Đoạn chương trình trên có quá nhiều lỗi:**
  1. **Lỗi về hàm `main()`:**
     - Cú pháp `void main()` không chuẩn trong C++. Theo chuẩn C++, hàm `main()` phải trả về kiểu `int`.
     - **Sửa:** Thay `void main()` bằng `int main()`.
  2. **Lỗi khi khởi tạo đối tượng `A a`:**
     - Lớp `A` không có constructor mặc định (không tham số). Lớp `A` chỉ có constructor với tham số `A(int t)`. Việc khai báo `A a;` sẽ gây lỗi vì thiếu tham số khi khởi tạo.
     - **Sửa:** Khi tạo đối tượng `a`, cần truyền vào một giá trị cho tham số `t`. Ví dụ:

```
        A a(10); // Khoi tao voi gia tri t = 10

```

3. **Lỗi khi gọi hàm `f2(a)`:**
   - Hàm `f2()` là phương thức thành viên của lớp `A`, không phải hàm toàn cục, nên không thể gọi `f2` ở bên ngoài lớp như khi gọi một hàm toàn cục.
   - `f2()` không nhận bất kỳ tham số nào, nhưng đoạn mã lại truyền vào đối tượng `a` như một tham số.
   - **Sửa:** Gọi phương thức `f2()` thông qua đối tượng `a`. Ví dụ:

```
        int result = a.f2(); // Neu muon lay gia tri tra ve
        cout << result;      // In ra gia tri cua x

```

   Hoặc, nếu chỉ cần gọi mà không cần sử dụng giá trị trả về:

```
        cout << a.f2();

```

4. **Lỗi trong phương thức `static void f()`:**
   - Phương thức tĩnh (`static`) không thể truy cập trực tiếp biến thành viên không tĩnh (`x`). Trong phương thức `static void f()`, việc sử dụng `cout « x;` sẽ gây lỗi vì `x` là biến thành viên của đối tượng.
   - **Sửa:** Loại bỏ từ khóa `static` trong khai báo phương thức `f()`, biến nó thành phương thức không tĩnh.

```
        void f() {
            cout << x;
        }

```

- **Code sau khi sửa:**

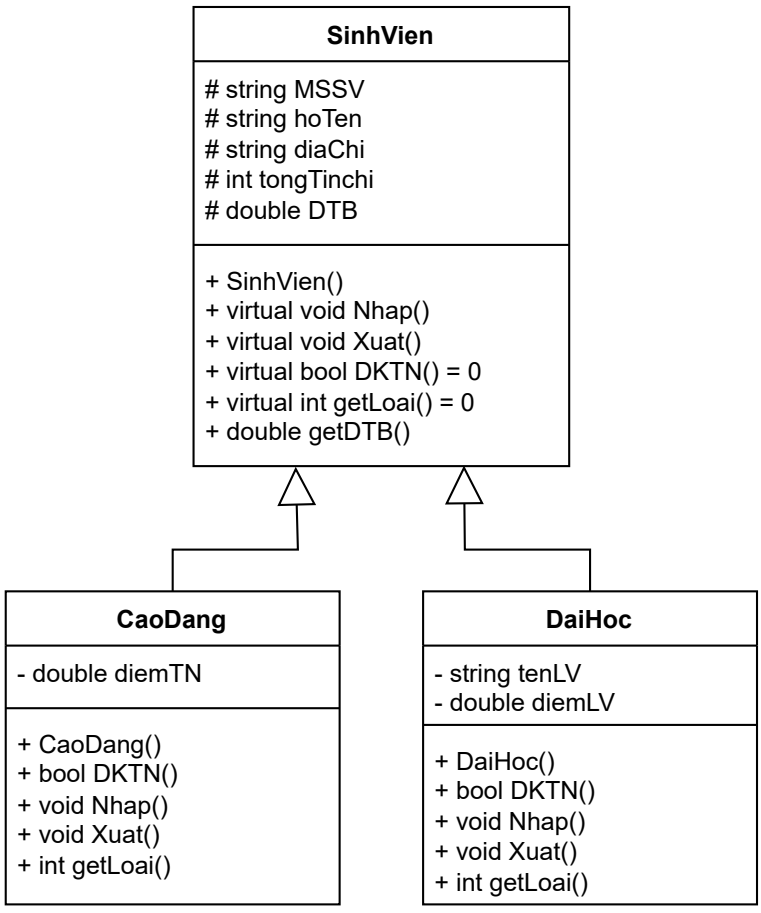
```
#include <iostream>
using namespace std;

class A {
 private:
    int x;
 public:
    A(int t) {
        x = t;
    }
    // Loai bo tu khoa static
    void f() {
        cout << x;
    }
    int f2() {
        return x;
    }
};

// Thay void main() bang int main()
int main() {
    // Khoi tao doi tuong voi tham so
    A a(10);
    // Goi phuong thuc f2() thong qua doi tuong a
    // In ra gia tri cua x, ket qua se in ra 10
    cout << a.f2();
    return 0;
}
```

1.3 Câu 3:

1.3.1 Đề bài:

- Giả sử Trường ĐH CNTT TP.HCM đào tạo sinh viên theo 2 hệ là hệ cao đẳng và hệ đại học. Thông tin cần quản lí của một sinh viên cao đẳng bao gồm: mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình, điểm thi tốt nghiệp. Thông tin cần quản lí của một sinh viên đại học bao gồm: mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình, tên luận văn, điểm luận văn. Cách xét tốt nghiệp của sinh viên mỗi hệ là khác nhau:
  - Sinh viên hệ cao đẳng tốt nghiệp khi có tổng số tín chỉ từ 120 trở lên, điểm trung bình từ 5 trở lên và điểm thi tốt nghiệp phải đạt từ 5 trở lên.
  - Sinh viên hệ đại học tốt nghiệp khi có tổng số tín chỉ từ 170 trở lên, điểm trung bình từ 5 trở lên và phải bảo vệ luận văn với điểm số đạt được từ 5 điểm trở lên.
- Bạn hãy đề xuất thiết kế các lớp đối tượng cần thiết để quản lý danh sách các sinh viên của Trường và hỗ trợ xét tốt nghiệp cho các sinh viên theo tiêu chí đặt ra như trên.
- Hãy viết chương trình bằng C++ cho phép thực hiện các yêu cầu sau:
  - Nhập vào danh sách sinh viên, có thể sử dụng string cho các chuỗi kí tự.
  - Cho biết số lượng sinh viên đủ điều kiện tốt nghiệp?
  - Cho biết sinh viên đại học nào có điểm trung bình cao nhất?

1.3.2 Class diagram:



1.3.3 Mô tả Input và Output bài toán:

Input: Nhập vào thông tin cho một danh sách sinh viên, bao gồm:

- Số lượng sinh viên.
- Với mỗi sinh viên, nhập loại sinh viên: cao đẳng hoặc đại học.
- Thông tin chi tiết của mỗi sinh viên:
  - Cao đẳng: mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình, điểm thi tốt nghiệp.
  - Đại học: mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình, tên luận văn, điểm luận văn.

  Trong đó: mã số sinh viên (chuỗi), họ tên (chuỗi), địa chỉ (chuỗi), tổng số tín chỉ (số nguyên), điểm trung bình (số thực), điểm thi tốt nghiệp (số thực), tên luận văn (chuỗi), điểm luận văn (số thực).

Output: Xuất kết quả là các thông tin hiển thị:

- Số lượng sinh viên đủ điều kiện tốt nghiệp theo các tiêu chí tương ứng với loại sinh viên:
  - Cao đẳng:
    - Tổng số tín chỉ ≥ 120.
    - Điểm trung bình ≥ 5.
    - Điểm thi tốt nghiệp ≥ 5.
  - Đại học:
    - Tổng số tín chỉ ≥ 170.
    - Điểm trung bình ≥ 5.
    - Điểm luận văn ≥ 5.
- Thông tin sinh viên đại học có điểm trung bình cao nhất, bao gồm: mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình, tên luận văn, điểm luận văn.

1.3.4 Mô tả hướng giải quyết bài toán:

Chia chương trình thành 7 file, trong đó:

- **File sinhvien.h, caodang.h và daihoc.h** khai báo lớp cơ sở SinhVien, các lớp dẫn xuất CaoDang và DaiHoc với các thuộc tính và phương thức đã nêu trong đề bài:
  - **File sinhvien.h: Xây dựng lớp cơ sở SinhVien:**
    - Thuộc tính chung: MSSV (mã số sinh viên), hoTen (họ tên), diaChi (địa chỉ), tongTinchi (tổng số tín chỉ), DTB (điểm trung bình).
    - Phương thức khởi tạo mặc định.
    - Các phương thức ảo nhập, xuất; các phương thức thuần ảo kiểm tra điều kiện tốt nghiệp, lấy loại sinh viên để hỗ trợ đa hình.
    - Phương thức truy xuất lấy điểm trung bình.
  - **File caodang.h: Xây dựng lớp dẫn xuất CaoDang:** thừa kế từ lớp SinhVien:
    - Thuộc tính bổ sung: diemTN (điểm tốt nghiệp).
    - Phương thức khởi tạo mặc định.
    - Các phương thức nhập, xuất thông tin của sinh viên cao đẳng.

    - Phương thức kiểm tra điều kiện tốt nghiệp của sinh viên cao đẳng.
    - Phương thức truy xuất lấy loại sinh viên cao đẳng.
  - **File daihoc.h: Xây dựng lớp dẫn xuất DaiHoc:** thừa kế từ lớp SinhVien:
    - Thuộc tính bổ sung: tenLV (tên luận văn), diemLV (điểm luận văn).
    - Phương thức khởi tạo mặc định.
    - Các phương thức nhập, xuất thông tin của sinh viên đại học.
    - Phương thức kiểm tra điều kiện tốt nghiệp của sinh viên đại học.
    - Phương thức truy xuất lấy loại sinh viên đại học.

- **File sinhvien.cpp, caodang.cpp và daihoc.cpp** định nghĩa nội dung phương thức trong các lớp SinhVien, CaoDang và DaiHoc:
  - **File sinhvien.cpp: Định nghĩa các phương thức cho lớp cơ sở SinhVien:**
    - Định nghĩa hàm khởi tạo mặc định với các thuộc tính nhận các giá trị mặc định.
    - Các phương thức nhập Nhap(), xuất Xuat() thông tin chung bao gồm mã số sinh viên, họ tên, địa chỉ, tổng số tín chỉ, điểm trung bình.
    - Phương thức truy xuất getDTB(): lấy ra điểm trung bình của sinh viên để tìm sinh viên đại học có điểm trung bình cao nhất.
    - Phương thức kiểm tra điều kiện tốt nghiệp và truy xuất loại sinh viên: không triển khai vì đây là phương thức ảo thuần tuý.
  - **File caodang.cpp và daihoc.cpp: Định nghĩa các phương thức cho lớp dẫn xuất CaoDang và DaiHoc:**
    - Định nghĩa hàm khởi tạo mặc định với các thuộc tính nhận các giá trị mặc định.
    - Phương thức nhập Nhap(): gọi Nhap() của SinhVien và nhập thêm điểm tốt nghiệp (đối với lớp CaoDang) hoặc tên luận văn, điểm luận văn (đối với lớp DaiHoc).
    - Phương thức xuất Xuat(): gọi Xuat() của SinhVien sau đó xuất thêm điểm tốt nghiệp (đối với lớp CaoDang) hoặc tên luận văn, điểm luận văn (đối với lớp DaiHoc).
    - Phương thức DKTN(): Kiểm tra điều kiện tốt nghiệp tương ứng với loại sinh viên:
      - Đối với lớp CaoDang: Nếu tổng số tín chỉ ≥ 120, điểm trung bình ≥ 5, điểm thi tốt nghiệp ≥ 5 thì trả về true (đủ điều kiện tốt nghiệp), ngược lại trả về false.
      - Đối với lớp DaiHoc: Nếu tổng số tín chỉ ≥ 170, điểm trung bình ≥ 5, điểm luận văn ≥ 5 thì trả về true (đủ điều kiện tốt nghiệp), ngược lại trả về false.
    - Phương thức truy xuất getLoai(): lấy loại của sinh viên, trong đó:
      - Đối với lớp CaoDang: trả về 1.
      - Đối với lớp DaiHoc: trả về 2.

- **File main.cpp** thực hiện gọi các phương thức trong hàm main():
  - Thực hiện nhập số lượng sinh viên.
  - Khởi tạo một vector chứa các con trỏ trỏ đến đối tượng kiểu SinhVien, dùng để quản lý cả sinh viên cao đẳng và sinh viên đại học trong cùng một vector nhờ cơ chế đa hình.
  - Thực hiện nhập danh sách sinh viên, với mỗi sinh viên:
    - Nhập loại sinh viên (Cao đẳng hoặc Đại học).
    - Tạo đối tượng tương ứng (CaoDang hoặc DaiHoc).
    - Gọi phương thức Nhap() của đối tượng để nhập thông tin.
    - Thêm đối tượng vào danh sách (vector).

  - Đếm số lượng sinh viên đủ điều kiện tốt nghiệp: Duyệt qua từng sinh viên trong danh sách, với mỗi sinh viên, gọi phương thức DKTN() để kiểm tra, nếu kết quả là true, tăng biến đếm số lượng.
  - Tìm sinh viên đại học có điểm trung bình cao nhất:
    * Khởi tạo biến điểm trung bình lớn nhất (maxDTB) bằng 0.
    * Duyệt qua danh sách, lọc các sinh viên là đối tượng DaiHoc bằng cách gọi phương thức getLoai().
    * Lấy điểm trung bình bằng phương thức getDTB() rồi so sánh với điểm trung bình lớn nhất (maxDTB) để tìm sinh viên có điểm cao nhất.
    * Duyệt qua lại danh sách, lọc các sinh viên là đối tượng DaiHoc, nếu có điểm trung bình bằng với điểm trung bình cao nhất thì thực hiện xuất thông tin sinh viên, nhằm in ra tất cả các sinh viên có cùng điểm trung bình cao nhất (nếu có).

- **Xuất kết quả:** Số lượng sinh viên đủ điều kiện tốt nghiệp và danh sách thông tin sinh viên đại học có điểm trung bình cao nhất.

1.3.5 Code:

- **Code file sinhvien.h:**

```
#include <iostream>
#include <string>
#include <iomanip>
#include <vector>
#pragma once
using namespace std;


// Lop co so SinhVien
class SinhVien {
 protected:
    string MSSV;        // Ma so sinh vien
    string hoTen;       // Ho ten sinh vien
    string diaChi;      // Dia chi sinh vien
    int tongTinchi;     // Tong so tin chi
    double DTB;         // Diem trung binh

 public:
    SinhVien();                 // Constructor mac dinh
    virtual void Nhap();        // Phuong thuc ao de nhap thong tin
    virtual void Xuat();        // Phuong thuc ao de xuat thong tin
    virtual bool DKTN() = 0;    // Phuong thuc ao thuan tuy de kiem tra dieu kien tot nghiep
    virtual int getLoai() = 0;  // Phuong thuc ao thuan tuy de lay loai sinh vien
    double getDTB();            // Phuong thuc lay diem trung binh
};
```

- **Code file caodang.h:**

```
#include "sinhvien.h"

// Lop CaoDang ke thua lop SinhVien
class CaoDang : public SinhVien {
 private:
    double diemTN;      // Diem thi tot nghiep cua sinh vien cao dang
 public:
    CaoDang();          // Constructor mac dinh
```

```
    bool DKTN();          // Phuong thuc kiem tra dieu kien tot nghiep cho sinh
    vien cao dang
    void Nhap();          // Phuong thuc nhap thong tin sinh vien cao dang
    void Xuat();          // Phuong thuc xuat thong tin sinh vien cao dang
    int getLoai();        // Phuong thuc tra ve loai sinh vien (loai cao dang)
};
```

- **Code file daihoc.h:**

```
#include "sinhvien.h"

// Lop DaiHoc ke thua lop SinhVien
class DaiHoc : public SinhVien {
 private:
    string tenLV;         // Ten luan van
    double diemLV;        // Diem luan van

 public:
    DaiHoc();             // Constructor mac dinh
    bool DKTN();          // Phuong thuc kiem tra dieu kien tot nghiep cho sinh
    vien dai hoc
    void Nhap();          // Phuong thuc nhap thong tin sinh vien dai hoc
    void Xuat();          // Phuong thuc xuat thong tin sinh vien dai hoc
    int getLoai();        // Phuong thuc tra ve loai sinh vien (loai dai hoc)
};
```

- **Code file sinhvien.cpp:**

```
#include "sinhvien.h"

// Constructor mac dinh
SinhVien::SinhVien() {}

/** Phuong thuc nhap thong tin cho sinh vien
    Input: Nhap ma so sinh vien (MSSV), ho ten, dia chi,
           tong so tin chi, diem trung binh*/
void SinhVien::Nhap() {
    cout << "Nhap ma so sinh vien: "; cin >> MSSV;
    cin.ignore();
    cout << "Nhap ho ten: "; getline(cin, hoTen);
    cout << "Nhap dia chi: "; getline(cin, diaChi);
    cout << "Nhap tong so tin chi: "; cin >> tongTinchi;
    cout << "Nhap diem trung binh: "; cin >> DTB;
}

/** Phuong thuc xuat thong tin sinh vien
    Output: Xuat thong tin cua sinh vien (MSSV, ho ten,
            dia chi, tong tin chi, diem trung binh) */
void SinhVien::Xuat() {
    cout << "Ma so sinh vien: " << MSSV << endl;
    cout << "Ho ten: " << hoTen << endl;
    cout << "Dia chi: " << diaChi << endl;
    cout << "Tong so tin chi: " << tongTinchi << endl;
    cout << "Diem trung binh: " << DTB << endl;
}

/** Phuong thuc lay diem trung binh cua sinh vien
    Output: Tra ve diem trung binh (DTB) cua sinh vien */
double SinhVien::getDTB() {
    return DTB;
}
```

- **Code file caodang.cpp:**

```cpp
#include "caodang.h"

// Constructor mac dinh cua lop CaoDang, ke thua tu SinhVien
CaoDang::CaoDang() {}

/** Phuong thuc kiem tra dieu kien tot nghiep cua sinh vien cao dang
    Output: Tra ve true neu sinh vien thoa man dieu kien tot nghiep, nguoc lai tra ve false
    Solution: Kiem tra tong tin chi >= 120, diem trung binh >= 5, diem thi tot nghiep >= 5 */
bool CaoDang::DKTN() {
    if (tongTinchi >= 120 && DTB >= 5 && diemTN >= 5) return true;
    return false;
}

/** Phuong thuc nhap thong tin cho sinh vien cao dang
    Input: Nhap ma so sinh vien, ho ten, dia chi, tong so tin chi,
           diem trung binh, diem thi tot nghiep
    Solution: Goi Nhap() cua SinhVien va nhap them diem tot nghiep */
void CaoDang::Nhap() {
    SinhVien::Nhap();
    cout << "Nhap diem thi tot nghiep: ";
    cin >> diemTN;
}

/** Phuong thuc xuat thong tin sinh vien cao dang
    Output: Xuat thong tin sinh vien cao dang (ma so, ho ten, dia chi,
            tong tin chi, diem trung binh, diem thi tot nghiep)
    Solution: Goi Xuat() cua SinhVien va xuat them diem tot nghiep */
void CaoDang::Xuat() {
    SinhVien::Xuat();
    cout << "Diem thi tot nghiep: " << diemTN << endl;
}

/** Phuong thuc tra ve loai sinh vien (1 cho sinh vien cao dang)
    Output: Tra ve gia tri 1, dai dien cho loai sinh vien cao dang */
int CaoDang::getLoai() {
    return 1;
}
```

- **Code file daihoc.cpp:**

```cpp
#include "daihoc.h"

// Constructor mac dinh cua lop DaiHoc, ke thua tu SinhVien
DaiHoc::DaiHoc() {}

/** Phuong thuc kiem tra dieu kien tot nghiep cua sinh vien dai hoc
    Output: Tra ve true neu sinh vien thoa man dieu kien tot nghiep, nguoc lai tra ve false
    Solution: Kiem tra tong tin chi >= 170, diem trung binh >= 5, diem luan van >= 5 */
bool DaiHoc::DKTN() {
    if (tongTinchi >= 170 && DTB >= 5 && diemLV >= 5) return true;
    return false;
}

/** Phuong thuc nhap thong tin cho sinh vien dai hoc
    Input: Nhap ma so sinh vien, ho ten, dia chi, tong so tin chi,
           diem trung binh, ten luan van, diem luan van
    Solution: Goi Nhap() cua SinhVien va nhap them ten luan van, diem luan van
    */
void DaiHoc::Nhap() {
```

```
    SinhVien::Nhap();
    cin.ignore();
    cout << "Nhap ten luan van: "; getline(cin, tenLV);
    cout << "Nhap diem luan van: "; cin >> diemLV;
}

/** Phuong thuc xuat thong tin sinh vien dai hoc
    Output: Xuat thong tin sinh vien dai hoc (ma so, ho ten, dia chi,
           tong tin chi, diem trung binh, ten luan van, diem luan van)
    Solution: Goi Xuat() cua Sinh Vien va xuat them ten luan van, diem luan van
     */
void DaiHoc::Xuat() {
    SinhVien::Xuat();
    cout << "Ten luan van: " << tenLV << endl;
    cout << "Diem luan van: " << diemLV << endl;
}

/** Phuong thuc tra ve loai sinh vien (2 cho sinh vien dai hoc)
    Output: Tra ve gia tri 2, dai dien cho loai sinh vien dai hoc */
int DaiHoc::getLoai() {
    return 2;
}
```

- **Code file main.cpp:**

```
#include "sinhvien.h"
#include "caodang.h"
#include "daihoc.h"

int main() {
    // Nhap so luong sinh vien
    int n;
    cout << "Nhap so luong sinh vien: "; cin >> n;
    vector<SinhVien*> dssv(n);

    // Cau 1: Nhap danh sach sinh vien
    bool coSVDH = false;
    for (int i = 0; i < n; i++) {
        // Loai sinh vien (1: Cao Dang, 2: Dai Hoc)
        int loai;
        cout << "Nhap loai sinh vien (1: Cao Dang, 2: Dai Hoc): ";
        cin >> loai;
        SinhVien *sv;
        if (loai == 1)
            sv = new CaoDang();
        else if (loai == 2) {
            sv = new DaiHoc();
            coSVDH = true;
        }
        // Nhap thong tin cho sinh vien
        sv->Nhap();
        // Luu doi tuong vao danh sach sinh vien
        dssv[i] = sv;
    }

    // Cau 2: Dem so luong sinh vien du dieu kien tot nghiep
    int cnt = 0;
    // Duyet qua danh sach sinh vien
    for (auto &sv : dssv)
        if (sv->DKTN()) cnt++;
    cout << "So luong sinh vien du dieu kien tot nghiep: " << cnt << endl;

    // Cau 3: Tim sinh vien dai hoc co diem TB cao nhat
```

```
    double maxDTB = 0;
    // Neu co sinh vien dai hoc
    if (coSVDH) {
        // Duyet qua danh sach sinh vien
        for (auto &sv: dssv)
            // Kiem tra loai sinh vien dai hoc va cap nhat diem trung binh cao nhat
            if (sv->getLoai() == 2 && sv->getDTB() > maxDTB)
                maxDTB = sv->getDTB();
        cout << "Sinh vien dai hoc co diem trung binh cao nhat:\n";
        // Duyet lai danh sach de xuat thong tin sinh vien co diem trung binh cao nhat
        for (auto &sv: dssv)
            if (sv->getLoai() == 2 && sv->getDTB() == maxDTB)
                sv->Xuat();
    }
    else cout << "Khong co sinh vien dai hoc trong danh sach.\n";

    // Giai phong bo nho
    for (auto &sv : dssv)
        delete sv;

    return 0;
}
```

- **Code đầy đủ có mô tả chi tiết input, output và solution của mỗi hàm/ phương thức:** Code Đề 1 - Câu 3.

1.3.6 Kiểm thử các test case:

- **Đầu vào mẫu:**

```
Nhap so luong sinh vien: 3
Nhap loai sinh vien (1: Cao Dang, 2: Dai Hoc): 2
Nhap ma so sinh vien: 23520036
Nhap ho ten: Cap Kim Hai Anh
Nhap dia chi: Hai Lang-Quang Tri
Nhap tong so tin chi: 175
Nhap diem trung binh: 10
Nhap ten luan van: Bai tap OOP
Nhap diem luan van: 10
Nhap loai sinh vien (1: Cao Dang, 2: Dai Hoc): 1
Nhap ma so sinh vien: 23520127
Nhap ho ten: Nguyen Thien Bao
Nhap dia chi: TP Dong Ha-Quang Tri
Nhap tong so tin chi: 161
Nhap diem trung binh: 9.5
Nhap diem thi tot nghiep: 9.75
Nhap loai sinh vien (1: Cao Dang, 2: Dai Hoc): 2
Nhap ma so sinh vien: 23520192
Nhap ho ten: Dang Quoc Cuong
Nhap dia chi: Gia Lai
Nhap tong so tin chi: 57
Nhap diem trung binh: 9.25
Nhap ten luan van: Bai tap review
Nhap diem luan van: 9.75
```

- **Kết quả đầu ra mẫu:**

```
So luong sinh vien du dieu kien tot nghiep: 2
Sinh vien dai hoc co diem trung binh cao nhat:
Ma so sinh vien: 23520036
```

```
Ho ten: Cap Kim Hai Anh
Dia chi: Hai Lang-Quang Tri
Tong so tin chi: 175
Diem trung binh: 10
Ten luan van: Bai tap OOP
Diem luan van: 10
```

2 Đề 20:

2.1 Câu 1:

2.1.1 Đề bài:

b.

- Phân biệt các phạm vi truy cập private, protected và public trong một lớp. Cho ví dụ về cách thức từ bên ngoài một lớp muốn truy cập các thành phần được thiết lập private trong lớp đó.
- Trình bày phương thức ảo là gì? Hãy cho biết trường hợp nào cần sử dụng đến loại phương thức này. Cho ví dụ minh hoạ.

2.1.2 Lời giải:

b.

- **Phân biệt các phạm vi truy cập private, protected và public trong một lớp:**
  - **Thuộc tính private:** Thành phần có thuộc tính private:
    - Là riêng tư của lớp đó
    - Chỉ có hàm thành phần của lớp và ngoại lệ các hàm bạn được phép truy xuất.
    - Các lớp con cũng không có quyền truy xuất.
  - **Thuộc tính protected:** Cho phép quy định một vài thành phần nào đó của lớp là bảo mật, theo nghĩa thế giới bên ngoài không được phép truy xuất, nhưng tất cả các lớp con, cháu... đều được phép truy xuất.
  - **Thuộc tính public:** Thành phần nào có thuộc tính public thì có thể truy xuất từ bất cứ nơi nào.
- **Ví dụ về cách thức từ bên ngoài một lớp muốn truy cập các thành phần được thiết lập private trong lớp đó:**
  - Sử dụng phương thức truy vấn để truy cập các thành viên private thông qua các hàm thành viên được định nghĩa trong lớp đó:

```cpp
#include <iostream>
using namespace std;

class Nguoi {
 private:
    string ten;
 public:
    Nguoi(string name) : ten(name) {}
    // Phuong thuc truy van truy cap thuoc tinh ten
    string getTen() {
        return ten;
    }
};

int main() {
    Nguoi nguoi("Cap Kim Hai Anh");
    // Truy cap cac thanh vien private thong qua cac ham cong khai
    cout << nguoi.getTen();
    return 0;
}
```

  - Sử dụng hàm bạn cho phép từ bên ngoài một lớp có thể truy xuất trực tiếp các thành private trong lớp đó:

```cpp
#include <iostream>
using namespace std;

class Nguoi {
 private:
    string ten;
 public:
    Nguoi(string name) : ten(name) {}
    // Su dung ham ban
    friend void Xuat(Nguoi &nguoi);
};

void Xuat(Nguoi &nguoi) {
    cout << nguoi.ten;
}

int main() {
    Nguoi nguoi("Cap Kim Hai Anh");
    // Truy cap truc tiep thong qua ham ban
    Xuat(nguoi);
    return 0;
}
```

- **Khái niệm phương thức ảo:** là phương thức của lớp cơ sở có thể được ghi đè trong các lớp dẫn xuất, là cách thể hiện tính đa hình trong ngôn ngữ C++.
- **Trường hợp cần sử dụng phương thức ảo:**
  - Các phương thức ở lớp cơ sở có tính đa hình phải được định nghĩa là một phương thức ảo
  - Khi muốn đảm bảo rằng phương thức phù hợp trong lớp dẫn xuất được gọi ngay cả khi sử dụng con trỏ hoặc tham chiếu kiểu lớp cơ sở.
- **Ví dụ minh hoạ:**

```cpp
#include <iostream>
using namespace std;

class Dongvat {
 public:
    // Phuong thuc ao
    virtual void Tiengkeu() {
        cout << "Tieng keu: " << endl;
    }
};

class Cho : public Dongvat {
 public:
    void Tiengkeu() {
        cout << "Cho keu: gau gau" << endl;
    }
};

class Meo : public Dongvat {
 public:
    void Tiengkeu() {
        cout << "Meo keu: meo meo" << endl;
    }
};

```

```
26 int main() {
27     // Tao cac doi tuong
28     Dongvat* dv1 = new Cho();
29     Dongvat* dv2 = new Meo();
30 
31     // Goi phuong thuc thong qua con tro lop co so
32     dv1->Tiengkeu(); // Goi phuong thuc tieng keu cua Cho
33     dv2->Tiengkeu(); // Goi phuong thuc tieng keu cua Meo
34 
35     // Giai phong bo nho
36     delete dv1;
37     delete dv2;
38 
39     return 0;
40     return 0;
41 }
```

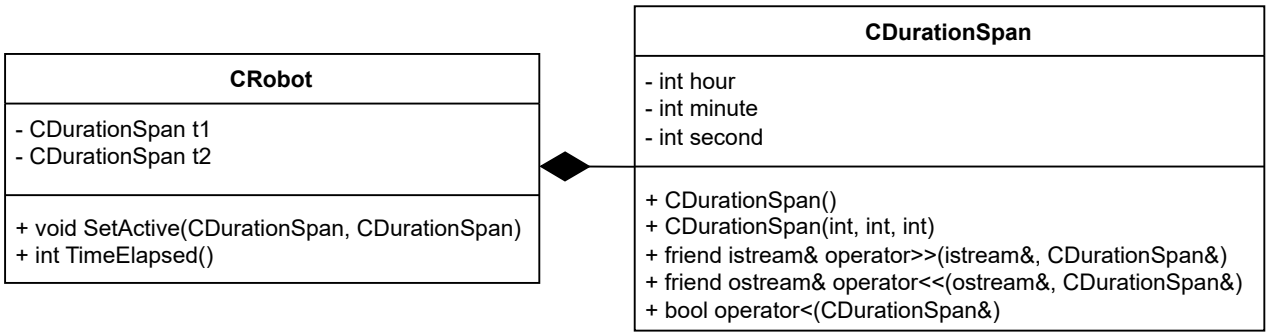
2.2 Câu 2:

2.2.1 Đề bài:

- Đề bài yêu cầu xây dựng chương trình quản lý thời gian hoạt động của một robot.
- Thời gian hoạt động của robot được tính bằng hiệu giữa thời gian bắt đầu và kết thúc. Cần thiết kế hai lớp:
  - **CDurationSpan**: Lớp biểu diễn khoảng thời gian với ba thuộc tính: giờ, phút, giây.
  - **CRobot**: Lớp đại diện cho robot, bao gồm hai đối tượng CDurationSpan: thời gian bắt đầu và thời gian kết thúc. Cần có phương thức tính thời gian hoạt động của robot.
- **Yêu cầu:** Định nghĩa các lớp CDurationSpan và lớp `CRobot` thích hợp để chương trình bên dưới không bị biên dịch lỗi và chạy đúng.

```
1  int main() {
2      CDurationSpan t1();
3      CDurationSpan t2(23, 55, 15);
4      CRobot r;
5  
6      cin >> t1;
7      cin >> t2;
8  
9      if (t1 < t2) {
10         r.SetActive(t1, t2);
11         cout << "Thoi gian Robot hoat dong la: " << r.TimeElapsed();
12     }
13     else {
14         cout << "Thiet lap thoi gian khong hop le!";
15     }
16 }
```

2.2.2 Class Diagram:



2.2.3 Mô tả Input và Output bài toán:

Input:

- Nhập vào thời gian bắt đầu và kết thúc cho robot, với định dạng (giờ phút giây).

Output:

- Xuất ra thời gian hoạt động của robot tính bằng giây. Nếu thời gian kết thúc nhỏ hơn thời gian bắt đầu, xuất thông báo lỗi.

2.2.4 Mô tả hướng giải quyết bài toán:

- **File CDurationSpan.h:**
  - Khai báo lớp CDurationSpan với các thuộc tính: giờ, phút, giây. Cung cấp các toán tử nhập và xuất.
  - Cung cấp phương thức so sánh thời gian (`operator<`) để so sánh hai đối tượng CDurationSpan.
- **File CRobot.h:**
  - Khai báo lớp CRobot với hai đối tượng CDurationSpan: thời gian bắt đầu và kết thúc.
  - Phương thức `SetActive` để thiết lập thời gian bắt đầu và kết thúc cho robot.
  - Phương thức `TimeElapsed` để tính toán thời gian hoạt động của robot.
- **File main.cpp:**
  - Nhập vào thời gian bắt đầu và kết thúc cho robot.
  - Kiểm tra nếu thời gian kết thúc nhỏ hơn thời gian bắt đầu, thông báo lỗi.
  - Nếu hợp lệ, tính và xuất thời gian hoạt động của robot.

2.2.5 Code:

- **Code file CDurationSpan.h:**

```
#include<bits/stdc++.h>
#pragma once

using namespace std;

class CDurationSpan {
private:
    int hour, minute, second;
    friend class CRobot;
```

```
public:
    CDurationSpan();
    CDurationSpan(int h, int m, int s);

    friend istream& operator >> (istream& is, CDurationSpan& d);
    friend ostream& operator << (ostream& os, const CDurationSpan& d);

    bool operator < (const CDurationSpan& d); // So sanh thoi gian
};
```

- **Code file CDurationSpan.cpp:**

```
#include<bits/stdc++.h>
#include "CDurationSpan.h"

using namespace std;

// Constructor mac dinh, khoi tao gio, phut, giay la 0
CDurationSpan::CDurationSpan() : hour(0), minute(0), second(0) {}

// Constructor co tham so
CDurationSpan::CDurationSpan(int h, int m, int s) : hour(h), minute(m), second(
    s) {}

// Toan tu nhap thoi gian
istream& operator >> (istream& is, CDurationSpan& d) {
    is >> d.hour >> d.minute >> d.second;
    return is;
}

// Toan tu xuat thoi gian
ostream& operator << (ostream& os, const CDurationSpan& d) {
    os << d.hour << " " << d.minute << " " << d.second;
    return os;
}

// Toan tu so sanh thoi gian
bool CDurationSpan::operator < (const CDurationSpan& d) {
    if (hour < d.hour) return true;
    if (hour == d.hour && minute < d.minute) return true;
    if (hour == d.hour && minute == d.minute && second < d.second) return true;
    return false;
}
```

- **Code file CRobot.h:**

```
#include<bits/stdc++.h>
#include "CDurationSpan.h"
#pragma once

using namespace std;

class CRobot {
private:
    CDurationSpan t1; // Thoi gian bat dau
    CDurationSpan t2; // Thoi gian ket thuc
public:
    void SetActive (CDurationSpan a, CDurationSpan b); // Thiet lap thoi gian
    bat dau va ket thuc
    int TimeElapsed(); // Tinh thoi gian hoat dong
};
```

- **Code file CRobot.cpp:**

```cpp
#include<bits/stdc++.h>
#include "CDurationSpan.h"
#include "CRobot.h"

using namespace std;

// Phuong thuc SetActive de thiet lap thoi gian bat dau va ket thuc
void CRobot::SetActive(CDurationSpan a, CDurationSpan b) {
    t1 = a;
    t2 = b;
}

// Phuong thuc TimeElapsed de tinh thoi gian hoat dong cua robot
int CRobot::TimeElapsed() {
    // Kiem tra neu t2 nho hon t1, tuc la t2 la thoi gian qua ngay
    int totalSecondsT1 = t1.hour * 3600 + t1.minute * 60 + t1.second;
    int totalSecondsT2 = t2.hour * 3600 + t2.minute * 60 + t2.second;

    // Neu t2 nho hon t1, cong them 24 gio cho t2 de tinh cho ngay tiep theo
    if (totalSecondsT2 < totalSecondsT1) {
        totalSecondsT2 += 24 * 3600;  // 24 gio = 86400 giay
    }

    return totalSecondsT2 - totalSecondsT1;
}
```

- **Code file main.cpp:**

```cpp
#include<bits/stdc++.h>
#include "CDurationSpan.h"
#include "CRobot.h"

using namespace std;

int main() {
    CDurationSpan t1;
    CDurationSpan t2(23, 55, 15);
    CRobot r;

    cout << "Nhap thoi gian bat dau (gio phut giay): ";
    cin >> t1;
    cout << "Nhap thoi gian ket thuc (gio phut giay): ";
    cin >> t2;

    // Kiem tra thoi gian ket thuc nho hon thoi gian bat dau
    if (t1 < t2) {
        r.SetActive(t1, t2);
        cout << "Thoi gian Robot hoat dong la: " << r.TimeElapsed() << " giay"
    << endl;
    } else {
        cout << "Thiet lap thoi gian khong hop le!" << endl;
    }

    return 0;
}
```

- **Code đầy đủ có mô tả chi tiết input, output và solution của mỗi hàm/ phương thức:** Code Đề 20 - Câu 2.

2.2.6 Kiểm thử các test case:

Test case 1: Trường hợp cơ bản 1:

```
1 0 0
2 0 0
Thoi gian Robot hoat dong la: 3600
Process returned 0 (0x0)   execution time : 2.534 s
```

Test case 2: Trường hợp cơ bản 2:

```
9 4 5
10 15 45
Thoi gian Robot hoat dong la: 4300
Process returned 0 (0x0)   execution time : 8.501 s
```

Test case 3: Thiết lập thời gian không hợp lệ:

```
15 10 15
15 10 14
Thiet lap thoi gian khong hop le!
Process returned 0 (0x0)   execution time : 5.590 s
```

2.3 Câu 3:

2.3.1 Đề bài:

Đề bài yêu cầu xây dựng chương trình quản lý thông tin xe, bao gồm các loại xe như xe xăng, xe lai xăng điện và xe điện. Mỗi xe có các thuộc tính chung như số khung, số máy, dung tích, màu sắc, tên hãng sản xuất, số chỗ ngồi, năm sản xuất, xuất xứ và giá nhập khẩu. Các loại xe sẽ có phương thức tính thuế tiêu thụ đặc biệt riêng, với các tỷ lệ khác nhau. Chương trình sẽ tính toán giá bán xe, phí đăng ký xe và thuế VAT.

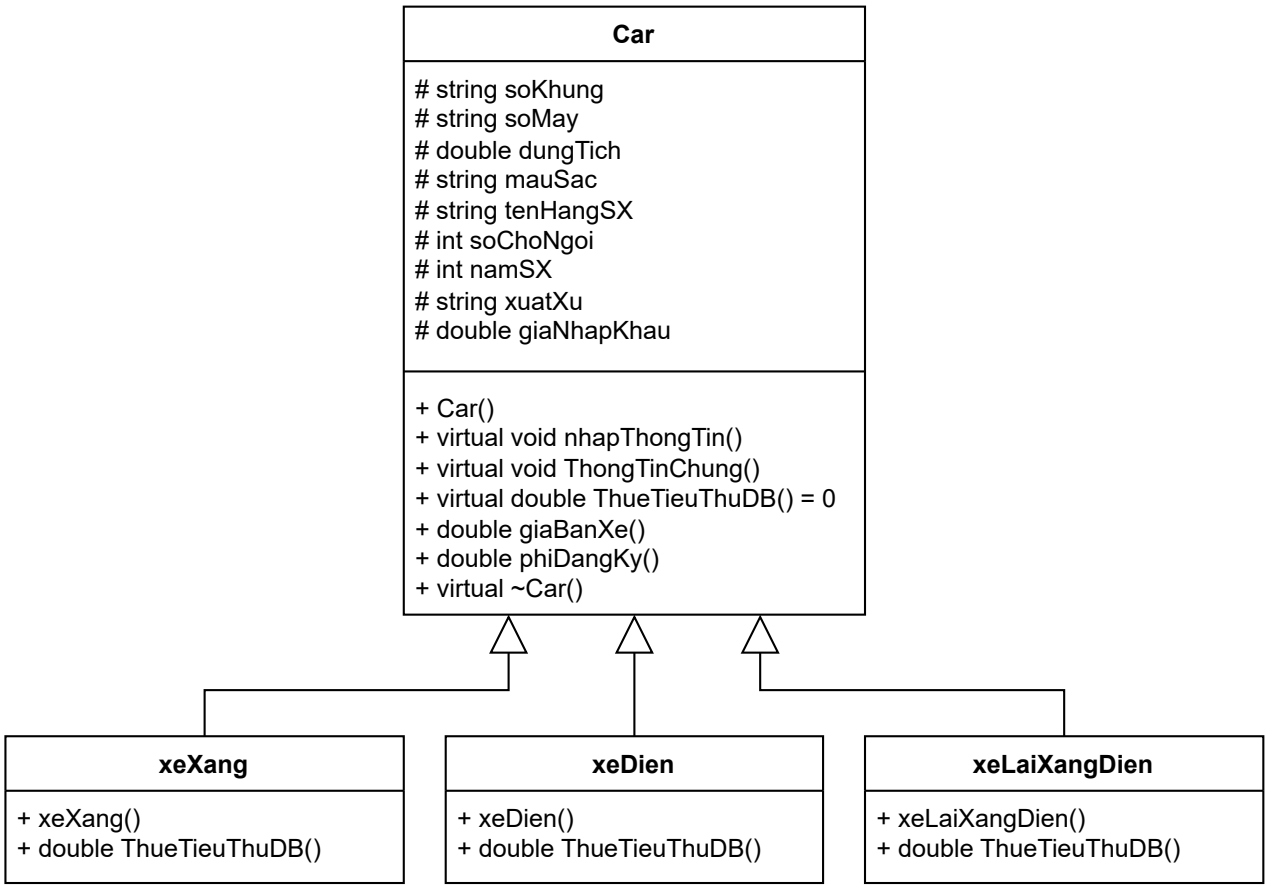
Chi phí chung của một chiếc xe được tính theo công thức sau :

Chi phí chung = Giá xe nhập khẩu + Thuế nhập khẩu + Thuế tiêu thụ đặc biệt + Biên lợi nhuận + thuế VAT + Chi phí đăng ký.

- Giá xe nhập khẩu là giá chiếc xe được nhập và vận chuyển về đến kho của công ty.
- Thuế nhập khẩu = 70% × Giá xe nhập khẩu.
- Thuế tiêu thụ đặc biệt được tính theo công thức:
  - Nếu là xe xăng Thuế tiêu thụ đặc biệt = 60% × (Giá xe nhập khẩu + Thuế nhập khẩu).
  - Nếu là xe điện Thuế tiêu thụ đặc biệt = 50% × (Giá xe nhập khẩu + Thuế nhập khẩu).
  - Nếu là xe xăng lai điện Thuế tiêu thụ đặc biệt = 55% × (Giá xe nhập khẩu + Thuế nhập khẩu).
- Biên lợi nhuận : Công ty để biên lợi nhuận 20% × (Giá xe nhập khẩu + Thuế nhập khẩu + Thuế tiêu thụ đặc biệt).
- Giá xe = Giá xe nhập khẩu + Thuế nhập khẩu + Thuế tiêu thụ đặc biệt + Biên lợi nhuận.
- Thuế VAT = 10% Giá xe.
- Giá bán = Giá xe + Thuế VAT.

  Khi khách hàng mua xe về để sử dụng cần phải thêm một khoản phí gọi là chi phí đăng ký, và được tính theo công thức sau:
- Chi phí đăng ký = 2% Giá xe.

2.3.2 Class Diagram:



2.3.3 Mô tả Input và Output bài toán:

Input:

- Nhập vào số lượng xe và loại xe (xe xăng, xe lai xăng điện, xe điện).
- Nhập thông tin chi tiết của từng xe.

Output:

- Xuất ra thông tin tổng số tiền của tất cả các xe.
- Xuất ra thông tin của xe có giá bán cao nhất.

2.3.4 Mô tả hướng giải quyết bài toán:

- **File Car.h:**
  - Khai báo lớp Car với các thuộc tính: số khung, số máy, dung tích, màu sắc, tên hãng sản xuất, số chỗ ngồi, năm sản xuất, xuất xứ và giá nhập khẩu.
  - Cung cấp các phương thức nhập thông tin, tính giá bán xe, tính phí đăng ký và thuế tiêu thụ đặc biệt.
  - Cung cấp phương thức ThongTinChung để xuất thông tin chi tiết của xe.
- **Các lớp xe con (xeDien, xeLaiXangDien, xeXang):**

  - Các lớp này kế thừa từ lớp Car và định nghĩa phương thức tính thuế tiêu thụ đặc biệt riêng cho từng loại xe.

- **File main.cpp:**
  - Nhập vào số lượng xe và loại xe.
  - Nhập thông tin chi tiết cho từng loại xe (xe xăng, xe lai xăng điện, xe điện).
  - Tính giá bán và phí đăng ký cho từng xe.
  - Tính tổng số tiền của tất cả các xe và tìm xe có giá bán cao nhất.

2.3.5 Code:

- **Code file Car.h:**

```
#include<bits/stdc++.h>
#pragma once
using namespace std;

// Lop co so Car chua cac thong tin chung ve xe
class Car {
protected:
    string soKhung;
    string soMay;
    double dungTich;
    string mauSac;
    string tenHangSX;
    int soChoNgoi;
    int namSX;
    string xuatXu;
    double giaNhapKhau;

public:
    // Constructor
    Car();
    virtual void nhapThongTin();
    virtual double ThueTieuThuDB() = 0;
    double giaBanXe();
    double phiDangKy();
    virtual void ThongTinChung();
    virtual ~Car();
};
```

- **Code file xeDien.h:**

```
#include "car.h"

class xeDien : public Car {
public:
    // Constructor
    xeDien();
    double ThueTieuThuDB();
};
```

- **Code file xeLaiXangDien.h:**

```
#include "car.h"

class xeLaiXangDien : public Car {
public:
    // Constructor
    xeLaiXangDien();
```

```
    double ThueTieuThuDB();
};
```

- **Code file xeXang.h:**

```
#include "car.h"

class xeXang : public Car {
public:
    // Constructor
    xeXang();
    double ThueTieuThuDB();
};
```

- **Code file Car.cpp:**

```
#include "Car.h"

Car::Car() {}

void Car::nhapThongTin() {
    cout << "Nhap so khung: "; cin >> soKhung;
    cout << "Nhap so may: "; cin >> soMay;
    cout << "Nhap dung tich: "; cin >> dungTich;
    cout << "Nhap mau sac: "; cin >> mauSac;
    cout << "Nhap ten hang SX: "; cin >> tenHangSX;
    cout << "Nhap so cho ngoi: "; cin >> soChoNgoi;
    cout << "Nhap nam SX: "; cin >> namSX;
    cout << "Nhap xuat xu: "; cin >> xuatXu;
    cout << "Nhap gia nhap khau: "; cin >> giaNhapKhau;
}

double Car::giaBanXe() {
    double thueNhapKhau = 0.7 * giaNhapKhau;
    double thueTieuThuDB = ThueTieuThuDB();
    double bienLoiNhuan = 0.2 * (giaNhapKhau + thueNhapKhau + thueTieuThuDB);
    double giaXe = giaNhapKhau + thueNhapKhau + thueTieuThuDB + bienLoiNhuan;
    double thueVAT = 0.1 * giaXe;
    double giaBan = giaXe + thueVAT;
    return giaBan;
}

double Car::phiDangKy() {
    return 0.02 * giaBanXe();
}

void Car::ThongTinChung() {
    cout << "So khung: " << soKhung << '\n';
    cout << "So may: " << soMay << '\n';
    cout << "Dung tich: " << dungTich << " cc\n";
    cout << "Mau sac: " << mauSac << '\n';
    cout << "Ten hang SX: " << tenHangSX << '\n';
    cout << "So cho ngoi: " << soChoNgoi << '\n';
    cout << "Nam SX: " << namSX << '\n';
    cout << "Xuat xu: " << xuatXu << '\n';
    cout << "Gia nhap khau: " << giaNhapKhau << " VND\n";
    cout << "Gia ban xe: " << giaBanXe() << " VND\n";
    cout << "Chi phi dang ky: " << phiDangKy() << " VND\n";
}

// Destructor
Car::~Car() {}
```

- **Code file xeDien.cpp:**

```cpp
#include "xeDien.h"

xeDien::xeDien() {}

double xeDien::ThueTieuThuDB() {
    return 0.5 * (giaNhapKhau + 0.7 * giaNhapKhau);
}
```

- **Code file xeLaiXangDien.cpp:**

```cpp
#include "xeLaiXangDien.h"

xeLaiXangDien::xeLaiXangDien() {}

double xeLaiXangDien::ThueTieuThuDB() {
    return 0.55 * (giaNhapKhau + 0.7 * giaNhapKhau);
}
```

- **Code file xeXang.cpp:**

```cpp
#include "xeXang.h"

xeXang::xeXang() {}

double xeXang::ThueTieuThuDB() {
    return 0.6 * (giaNhapKhau + 0.7 * giaNhapKhau);
}
```

- **Code file main.cpp:**

```cpp
#include<bits/stdc++.h>
#include "car.h"
#include "xeXang.h"
#include "xeLaiXangDien.h"
#include "xeDien.h"

using namespace std;

int main() {
    // Danh sach xe
    vector<Car*> cars;
    int n;
    cout << "Nhap so luong xe: "; cin >> n;

    /*
        Ta co cac loai xe
        1. Xe xang
        2. Xe lai xang dien
        3. Xe dien
    */

    for (int i = 1; i <= n; i++) {
        cout << "Nhap thong tin xe thu " << i << ":\n";
        cout << "Nhap loai xe: ";
        int loai;
        cin >> loai;
        Car* a;
        if (loai == 1)
            a = new xeXang();
        else if (loai == 2)
            a = new xeLaiXangDien();
```

```
32          else a = new xeDien();
33          a->nhapThongTin();
34          cars.push_back(a);
35      }
36
37      double tongSoTien = 0.0;
38      for (auto it : cars)
39          tongSoTien += it->giaBanXe();
40      cout << "Tong so tien: " << tongSoTien << '\n';
41
42      Car* maxCar = nullptr;
43      for (auto it : cars) {
44          if (maxCar == nullptr || maxCar->giaBanXe() < it->giaBanXe()) {
45              maxCar = it;
46          }
47      }
48
49      if (maxCar != nullptr) {
50          cout << "Thong tin xe co gia cao nhat: " << '\n';
51          maxCar->ThongTinChung();
52      }
53
54      // Giai phong bo nho
55      for (Car* car : cars)
56          delete car;
57
58      return 0;
59
60  }
```

- **Code đầy đủ có mô tả chi tiết input, output và solution của mỗi hàm/ phương thức:** Code Đề 20 - Câu 3.

2.3.6 Kiểm thử các test case

- **Đầu vào mẫu:**

```
1  Nhap so luong xe: 3
2  Nhap thong tin xe thu 1
3  1
4  Nhap so khung: AB1234
5  Nhap so may: XY5678
6  Nhap dung tich: 2000
7  Nhap mau sac: Do
8  Nhap ten hang SX: Toyota
9  Nhap so cho ngoi: 5
10 Nhap nam SX: 2020
11 Nhap xuat xu: Japan
12 Nhap gia nhap khau: 500000000
13
14 Nhap thong tin xe thu 2
15 2
16 Nhap so khung: CD1234
17 Nhap so may: XY8765
18 Nhap dung tich: 1800
19 Nhap mau sac: Xanh
20 Nhap ten hang SX: Honda
21 Nhap so cho ngoi: 2021
22 Nhap nam SX: 2021
23 Nhap xuat xu: VietNam
24 Nhap gia nhap khau: 450000000
25
```

```
Nhap thong tin xe thu 3
3
Nhap so khung: EF1234
Nhap so may: AB818
Nhap dung tich: 1000
Nhap mau sac: Tim
Nhap ten hang SX: Vinfast
Nhap so cho ngoi: 5
Nhap nam SX: 2025
Nhap xuat xu: VietNam
Nhap gia nhap khau: 100000000
```

- **Kết quả đầu ra mẫu:**

```
Tong so tien: 3696990000.00000
Thong tin xe co gia cao nhat:
So khung: AB1234
So may: XY5678
Dung tich: 2000.00000 cc
Mau sac: Do
Ten hang SX: Toyota
So cho ngoi: 5
Nam SX: 2020
Xuat xu: Japan
Gia nhap khau: 500000000.00000 VND
Gia ban xe: 1795200000.00000 VND
Chi phi dang ky: 35904000.00000 VND
```

3 Phụ lục:

- **Danh sách Class diagram:** Class diagram.
- **Danh sách Code:** Code.
- **Bài thuyết trình:** Slide.
- **Video thuyết trình:** Link video.