Задача 1

Дадени са сигнатурите на няколко класа. Има ли грешка в дизайна на класовете, и ако да, каква е тя ? Обяснете с 1-2 изречения каква е грешката, и защо избрания подход е грешен.

Бележка: Приемете, че няма имплементационни грешки. Приемете, че някои член-данни нямат нужда от променяне, извън конструктора.

```
class Vehicle{
public:
    Vehicle() = default;
    Vehicle(const std::string& init_maker, const std::string& init_model, const
std::string& init_engine, const unsigned& init_horsepower);

    std::string get_maker() const;
    std::string get_model() const;
    std::string get_engine() const;
    unsigned get_horsepower() const;

    void set_engine(const std::string& new_engine);
    void set_horsepower(const unsigned& new_horsepower);
private:
    std::string maker;
    std::string model;
    std::string engine;
    unsigned horsepower;
};

class SteeringWheel{
public:
    SteeringWheel() = default;
    SteeringWheel(const unsigned& init_model_number, const unsigned&
init_radius);

    unsigned get_model_number() const;
    unsigned get_radius() const;

private:
    unsigned model_number;
    unsigned radius;
};

class Car: public Vehicle, public SteeringWheel{
public:
    Car();
    Car(const std::string& init_chasiss_type);

    std::string get_chassis_type() const;
private:
    std::string chassis_type;
};
```

Задача 2

Какво ще изведе на конзолата следния код ? Вярно ли работи програмата ? Обяснете защо.

```
#include <iostream>

class A{
public:
    void speak(){
        std::cout << "Foo" << std::endl;
    }
};

class B: public A{
public:
    void speak(){
        std::cout << "Bar" << std::endl;
    }
};

int main(){

    B temp;
    temp.speak();

    A* ptr_to_temp = &temp;
    ptr_to_temp->speak();

    return 0;
}
```

Задача 3

Не е позволено използването на STL

Бележка: Приемаме, че два обекта са "еднакви", когато резултатът от изпълнението на оператор= върху тях е `true`

Напишете клас `UniqueBox`, който пази в себе си "различни" променливи, от тип, зададен от потребителя.

Добавянето на елемент към `UniqueBox` трябва да се случва само чрез оператора `+=`. Освен това, при опит за добавяне на нов елемент, се прави проверка дали този елемент вече съществува в колекцията, т.е. в един `UniqueBox` не можем да пазим два "еднакви" обекта.

За достъп до елемент от `UniqueBox` използваме оператор `[]`.

Класът `UniqueBox` трябва да притежава и метод, който да показва колко елемента имаме в колекцията към момента.

Примерна програма:

```
int main(){
    UniqueBox<int> container;

    container += 3;
```

```
    container += 5;

    std::cout << container.get_counter() << std::endl;

    container += 6;
    container += 3;

    std::cout << container.get_counter() << std::endl;

    std::cout << container[1] << std::endl;

    return 0;
}
```

Примерен изход:

```
2
3
5
```

Задача 4

Позволено е използването на STL

Напишете програма, която да обработва поръчки в магазин.

Магазинът трябва да поддържа следните видове продукти:

Хранителен продукт:

- Име
- Код (низ)
- Цена
- Калории

Напитка:

- Име
- Код (низ)
- Цена
- Алкохолно съдържание

Магазинът трябва да поддържа следните функционалности:

- Добавяне на нови продукти в магазина
- Премахване на продукти от магазина (по код)
- Извеждане на всички продукти, които са с по-ниска цена от такава, въведена от потребителя
- Търсене на продукт по код