Задачи:

1. Създайте клас Stock, който представлява финансов инструмент (акция). Класът да съдържа следните член-данни: abbreviation – абревиатура на акцията – статичен масив с до 5 символа, name - име на акцията - статичен масив от максимум 20 символа, prices - масив, който съхранява цената на акцията през последните 12 месеца. Реализирайте следните функции: гетъри и сетъри за всички член-данни., конструктор по подразбиране, конструктор с параметри копиращ конструктор, оператор =. Реализрайте isProfitable: функция, която проверява дали акцията е печеливша. Тя е печеливша, ако през последните 12 месеца стойността ѝ е нараствала повече месеци в сравнение с тези, в които е намалявала, и ако крайната ѝ цена е с поне 10% по-висока в сравнение с цената ѝ в началото на периода. Използвайте разделна компилация компилация за решението на задачата.

2. Създайте клас Car, който представлява превозно средство. Класът да съдържа следните член-данни: regNumber – регистрационен номер на превозното средство, статичен масив от 8 символа, manufacturer - марка на превозното средство - статичен масив от максимум 20 символа, model - модел на превозното средство, което е статичен масив от максимум 20 символа, fuelConsumption - масив с три стойности, съхраняващ горивната консумация на превозното средство за градско, извънградско и комбинирано шофиране. Реализирайте следните функции: гетъри и сетъри за всички член-данни, конструктор по подразбиране, конструктор с параметри, копиращ конструктор, оператор =. Реализирайте isFuelEfficient: функция, която проверява дали превозното средство е икономично. Превозното средство се счита за икономично, ако средната му горивна консумация за градско, извънградско и комбинирано шофиране не надвишава 6 литра на 100 км. Реализирайте функция isMoreFuelEfficient: функция, която приема друго превозно средство и проверява дали текущото превозно средство е по-икономично от подаденото. Използвайте разделна компилация компилация за решението на задачата.

3. Реализирайте клас Engine. Всеки двигател има: power - мощност цяло число, displacement – обем в кубически милиметри цяло число, VIN – номер на шаси статичен масив от символи с дължина 17 символа. Двигателят е специфичен за автомобила (т.е. не може да се използва в друг автомобил). Реализирайте клас Wheel – джанта. Нека тя има следните член данни: radius – цяло число в инчове, material – изброим тип с 4 възможни стойности – Steel, Aluminium, Carbon, Magnesium. Джантата може да се използва и в други автомобили или велосипеди, камиони и т.н. В класа кола от предната задача добавете: двигател (обект от клас Engie) и джанта(обект от клас Wheel).

Всяка кола "притежава" конкретен двигател, така че това е пример за композиция. В същото време, колата използва джанти, които не са уникални за нея и могат да се използват от други автомобили, така че това е пример за агрегация.

4. Разработка на система за симулация на състезание с автомобили.
   Към класът Vehicle от предните две задачи добавете следните член данни:
   Drivetrain: Вид задвижване – изброим тип с три стойности предно, задно или 4x4
   Acceleration: ускорение, цяло число от 0 до 100
   TopSpeed: максимална скорост, цяло число от 0 до 100
   Handling: управление, цяло число от 0 до 100
   Реализирайте клас писта Track, който има следните елементи:
   SlowCorners - брой остри завои: цяло число
   FastCorners - брой тъпи завои: цяло число
   StarightLength - дължина на правата част в метри: цяло число
   Клас състезание Race, който има следните елементи:
   Track (обект от клас Писта)
   Cars (списък от обекти от клас Кола)
   В Състезание трябва да има функция classify(), която принтира автомобилите според техния ред на финиширане. Редът на финиширане се определя по следния начин: броят на острите завои на пистата се умножава по ускорението на колата, и ако колата е с 4x4 задвижване, резултатът се умножава по коефициент 1.2. Броят на тъпите завои се умножава по способността на колата за управление и ако колата е с предно задвижване, резултатът се умножава по коефициент 0.9. Дължината на правата в метри се дели на 100 и се умножава по максималната скорост на колата. Всички тези стойности се събират, за да се получи общ резултат за всяка кола. Колите се подреждат според общия им резултат, като колата с най-малък резултат финишира първа.
   Използвайте разделна компилация компилация за решението на задачата.

5. Разработка на система за управление на кинофестивал: Създайте клас филм Movie Всеки филм има: заглавие title - статичен масив с до 30 символа, режисьор director - статичен масив с до 30 символа, продължителност length – цяло число. Създайте клас прожекция Projection, съдържаща в себе си филм (обект от клас Movie), дата и час на прожекцията datetime – сатичен масив с дължина 15 и име на зала hallName– статичен масив с максимална дължина 20. Създайте клас кинофестивал Festival, който съдържа: име на фестивала festivalName - статичен масив с до 30 символа ,списък от прожекции (списък от обекти от клас Прожекция).
   Използвайте разделна компилация компилация за решението на задачата.

В тази система, всеки кинофестивал се състои от множество прожекции, а всяка прожекция включва по един филм. Това е пример за композиция, тъй като прожекцията не може да съществува без филма, и ако прожекцията бъде премахната, филмът, свързан с нея, също ще престане да бъде част от фестивала.