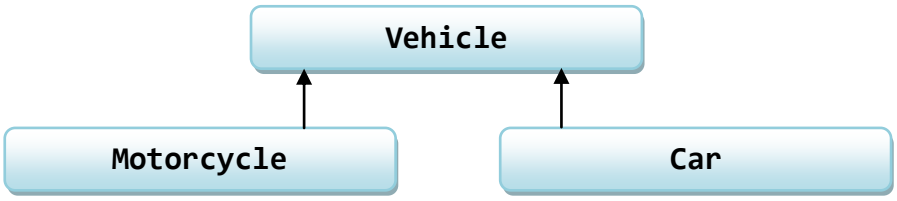
Zadatak:

Napišite sučelje i implementaciju za klase **`Vehicle, Motorcycle i Car`**. Sljedeći dijagram opisuje relacije među klasama. Sučelja za klase spremite u datoteku **`vehicle.h`**, a implementacije u datoteku **`vehicle.cpp`**.



Klasa `Vehicle`

Ova klasa opisuje vozilo. Svako vozilo ima tri parametra koji ga određuju: maksimalna brzina, broj kubika i registracija. Vozilo je brže što mu je maksimalna brzina veća. Ako dva vozila imaju jednaku maksimalnu brzinu brže je ono s većim broj kubika. Možete pretpostaviti da u programu neće postojati dva vozila s istom maksimalnom brzinom i istim brojem kubika.

- **`Vehicle(int max_speed, int cc, string registration)`**
  Konstruktor za klasu vozilo.

- **`int get_max_speed()`**
  Vraća maksimalnu brzinu vozila.
  .
- **`int get_cc()`**
  Vraća broj kubika.

- **`string get_registration()`**
  Vraća registraciju vozila.

- **`int get_gas();`**
  Vraća postotak goriva u vozilu. Svako kreirano vozilo na početku ima 100% pun rezervoar.

- **`void drive_10km()`**
  Simulira vožnju vozila. Nakon poziva ove funkcije rezervoar je za 10% prazniji (10% od ukupnog kapaciteta).

- **`list<Vehicle *> same_speed();`**
  Vraća listu pokazivača na sva vozila (gledaju se sva vozila trenutno kreirana u programu) koja imaju jednaku maksimalnu brzinu kao i vozilo koje je pozvalo funkciju. Lista mora biti sortirana prema brzini vozila.

- **`static Vehicle& fastest();`**
  Vraća referencu na najbrže vozilo (s obzirom na sva trenutno kreirana vozila u programu).

Klasa `Motorcycle`

Ova klasa opisuje motor. Motor ima dodatni parametar koji govori o vrsti motora (skuter ili trkaći motor).

- **`Motorcycle(int max_speed, int cc, string registration, char type)`**
  Konstruktor za klasu motor. Varijabla type je jednaka 'R' za trkaći motor, a 'S' za skuter.

- **`void drive_10km()`**
  Simulira vožnju motora. Nakon poziva ove funkcije rezervoar je za 5% prazniji (5% od ukupnog kapaciteta).

- **`string get_type();`**
  Vraća „race“ ako je trkaći motor, a „skuter“ ako je skuter.

Klasa `Car`

Ova klasa opisuje automobil. Automobil ima dodatni parametar za volumen prtljažnika.

- **`Car(int max_speed, int cc, string registration, int volume)`**
  Konstruktor za klasu automobil.

- **`void drive_10km()`**
  Simulira vožnju automobila. Nakon poziva ove funkcije rezervoar je za 15% prazniji (15% od ukupnog kapaciteta).

- **`int get_volume()`**
  Vraća kapacitet prtljažnika.

Primjer klijentskog programa

```
#include <iostream>
#include <list>
#include "vehicle.h"

int main(void)
{
    Vehicle A(200, 500, "AAAA"), B(200, 600, "BBBB"), C(240, 1100, "CCCC");
    Motorcycle M(300, 600, "MMMM", 'R');
    Car T(150, 500, "TTTT", 100);
    Vehicle *niz_vozila[5];
    niz_vozila[0] = new Car(200, 900, "R0000", 70);
    niz_vozila[1] = new Motorcycle(300, 650, "R1111", 'R');
    niz_vozila[2] = new Vehicle(250, 800, "R2222");
    niz_vozila[3] = new Car(150, 400, "R3333", 90);
    niz_vozila[4] = new Motorcycle(200, 300, "R4444", 'S');

    cout << "Motor M:" << endl;
    cout << M.get_max_speed() << " "
         << M.get_cc() << " "
         << M.get_registration() << " "
         << M.get_type() << endl;
    // Motor M:
    // 300 600 MMMM race

    for(int i = 0; i < 5; i++)
    {
        niz_vozila[i]->drive_10km();
        niz_vozila[i]->drive_10km();
        niz_vozila[i]->drive_10km();
    }

    cout << "Nakon voznje:" << endl;
    for(int i = 0; i < 5; i++)
        cout << niz_vozila[i]->get_max_speed() << " "
             << niz_vozila[i]->get_cc() << " "
             << niz_vozila[i]->get_registration() << " "
             << niz_vozila[i]->get_gas() << endl;
    // Nakon voznje:
    // 200 900 R0000 55
    // 300 650 R1111 85
    // 250 800 R2222 70
    // 150 400 R3333 55
    // 200 300 R4444 85

    cout << "Najbrzi:" << endl;
    Vehicle &najbrzi = Vehicle::fastest();
    cout << najbrzi.get_max_speed() << " "
         << najbrzi.get_cc() << " "
         << najbrzi.get_registration() << endl;
    //Najbrzi:
    //300 650 R1111

    cout << "Jednaka brzina:" << endl;
    list<Vehicle *> lista_vozila = A.same_speed();
    list<Vehicle *>:: iterator it;
    for(it = lista_vozila.begin(); it != lista_vozila.end(); ++it)
        cout << (*it)->get_cc() << " " << (*it)->get_registration() << endl;
    // Jednaka brzina:
    // 300 R4444
    // 500 AAAA
    // 600 BBBB
    // 900 R0000

    for(int i = 0; i < 5; i++)
        delete niz_vozila[i];

    return 0;
}
```

Opće napomene

- Klase, funkcije i datoteke koje šaljete moraju se zvati točno onako kako je zadano u zadatku. Pazite na mala i velika slova!
- Trebate poslati samo sučelje i implementaciju. U datotekama koje šaljete ne smije se nalaziti funkcija main()!
- nijedna funkcija ne smije ništa učitavati s tipkovnice ili neke datoteke, niti išta ispisivati na ekran ili u neku datoteku.
- Svaki od main-ova pomoću kojih testiramo ispravnost vašeg programa neće pozivati sve gore navedene funkcije. Stoga, ako neku od funkcija ne znate napisati ipak možete dobiti koji bod (u tom slučaju tu funkciju nemojte navesti niti u .h niti u .cpp datoteci ili napravite neku trivijalnu implementaciju).

Ispravnost implementacija koje napišete bit će provjerena tako da ćemo mi napisati razne klijentske programe koji će deklarirati nekoliko varijabli zadane strukture, i na njima pozivati funkcije koje ste trebali napisati. Ako se poslani programi ne budu uspješno povezivali (linkali) s našim klijentskim programima, smatrat će se neispravnima.

Neki klijentski programi provjeravat će samo neke jednostavnije funkcije, dok će neki provjeravati sve funkcije koje trebate napisati. Provjera je potpuno automatska, tako da je od presudne važnosti da se pridržavate specifikacije. Nepridržavanje lako može uzrokovati osvojenih 0 bodova iz zadaće!

Naravno, za provjeru radi li implementacija prije nego što je pošaljete, preporučuje se da je testirate pomoću nekog klijentskog programa. No taj klijentski program ne šaljete!

Za sva pitanja vezana uz ovu zadaću javite se asistentu Vujčiću na jvujcic+rp1@gmail.com.