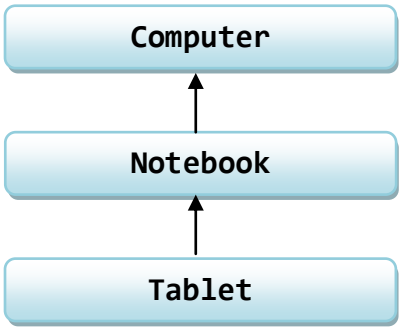
Zadatak:

Napišite sučelje i implementaciju za klase **`Computer, Notebook i Tablet`**. Sljedeći dijagram opisuje relacije među klasama. Sučelja za klase spremite u datoteku **`comp.h`**, a implementacije u datoteku **`comp.cpp`**.



Klasa `Computer`

Ova klasa opisuje računalo. Svako računalo ima tri parametra koji ga određuju: brzina procesora (MHz), proizvođač procesora i količina memorije (MB). Računalo je brže ako ima brži procesor. U slučaju da je brzina procesora jednaka, kažemo da je brže računalo ono s više memorije. U testnim primjerima nikad neće biti dva računala s jednakom brzinom procesora i jednakom količinom memorije.

- **`Computer(int mhz, string vendor, int memory)`**
  Konstruktor za klasu Computer.

- **`int get_mhz()`**
  Vraća brzinu procesora.
  .
- **`int get_memory()`**
  Vraća količinu memorije.

- **`string get_vendor()`**
  Vraća marku procesora.

- **`int upgrade(int mhz, int memory)`**
  *Nadograđuje računalo s novim procesorom brzine mhz i novom memorijom količine memory. Vraća cijenu nadogradnje računala prema formuli: (nova_brzina – stara_brzina)*2 + (nova_memorija – stara_memorijy)*1.*

- **`Computer& next_fastest()`**
  Vraća referencu na iduće po redu brže računalo (s obzirom na sva trenutno kreirana računala u programu).

- **`static Computer& fastest_computer()`**
  Vraća referencu na najbrže računalo (s obzirom na sva trenutno kreirana računala u programu).

- **`static Computer& faster(Computer &comp1, Computer &comp2);`**
  Vraća referencu na brže od dva proslijeđena računala.

Klasa `Notebook`

Ova klasa opisuje notebook. Notebook ima i parametar veličine ekrana.

- **`Notebook(int mhz, string vendor, int memory, float screen_size)`**
  Konstruktor za klasu Notebook.

- **`float get_screen_size()`**
  Vraća veličinu ekrana.

- **`int upgrade(int mhz, int memory)`**
  *Nadograđuje notebook s novim procesorom brzine mhz i novom memorijom količine memory. Vraća cijenu nadogradnje računala prema formuli: (nova_brzina – stara_brzina)*3 + (nova_memorija – stara_memorijy)*2.*

Klasa `Tablet`

Ova klasa opisuje Tablet. Tablet ima i parametar OS-a.

- **`Tablet(int mhz, string vendor, int memory, float screen_size, string os)`**
  Konstruktor za klasu Tablet.

- **`string get_os()`**
  Vraća naziv operativnog sustava.

- **`int upgrade(int mhz, int memory)`**
  *Nadograđuje tablet s novim procesorom brzine mhz i novom memorijom količine memory. Vraća cijenu nadogradnje računala prema formuli: (nova_brzina – stara_brzina)*3 + (nova_memorija – stara_memorijy)*3.*

Primjer klijentskog programa

```
#include <iostream>
#include "comp.h"

int main(void)
{
    Computer A(1800, "intel", 1024), B(2500, "amd", 23), C(2200, "intel", 4096);
    Computer *niz_racunala[5];
    niz_racunala[0] = new Notebook(1500, "intel", 2048, 15.6);
    niz_racunala[1] = new Computer(3300, "apple", 1024);
    niz_racunala[2] = new Tablet(2000, "nvidia", 2048, 15.6, "android");
    niz_racunala[3] = new Tablet(1700, "arm", 2048, 10.1, "ios");
    niz_racunala[4] = new Notebook(2000, "via", 4096, 17.3);

    Computer &brzi = Computer::fastest_computer();
    cout << "Najbrze racunalo:" << endl;
    cout << brzi.get_mhz() << " " << brzi.get_vendor() << " " << brzi.get_memory() << endl;
    // Najbrze racunalo:
    // 3300 apple 1024

    cout << "Sljedece najbrze racunalo:" << endl;
    Computer &next = A.next_fastest();
    cout << next.get_mhz() << " " << next.get_vendor() << " " << next.get_memory() << endl;
    // Sljedece najbrze racunalo:
    // 2000 nvidia 2048

    cout << "Cijene nadogradnji:" << endl;
    for(int i = 0; i < 5 ; i++)
        cout << niz_racunala[i]->upgrade(niz_racunala[i]->get_mhz() + 1000,
                                         niz_racunala[i]->get_memory() + 1024) << endl;
    // Cijene nadogradnji:
    // 5048
    // 3024
    // 6072
    // 6072
    // 5048

    cout << "Nadogradjena racunala:" << endl;
    for(int i = 0; i < 5 ; i++)
        cout << niz_racunala[i]->get_mhz() << " "
             << niz_racunala[i]->get_vendor() << " "
             << niz_racunala[i]->get_memory() << endl;
    // Nadogradjena racunala:
    // 2500 intel 3072
    // 4300 apple 2048
    // 3000 nvidia 3072
    // 2700 arm 3072
    // 3000 via 5120

    Tablet t(1500, "intel", 1024, 9.8, "android");
    cout << "Tablet" << endl;
    cout << t.get_mhz() << " "
         << t.get_vendor() << " "
         << t.get_memory() << " "
         << t.get_screen_size() << " "
         << t.get_os() << endl;
    //Tablet
    //1500 intel 1024 9.8 android

    Computer X(1800, "amd", 1024), Y(2200, "via", 1024);
    Computer &Z = Computer::faster(X, Y);
    cout << "Brze racunalo:" << endl;
    cout << Z.get_mhz() << " " << Z.get_vendor() << " " << Z.get_memory() << endl;
    //Brze racunalo:
    //2200 via 1024

    for(int i = 0; i < 5; i++)
        delete niz_racunala[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.