Zadaci za Laboratorijsku vježbu 9

NAPOMENA: Studenti bi trebali da razmisle o zadacima koji će se raditi na laboratorijskoj vježbi prije nego što dođu na vježbu, tako da već u startu imaju osnovne ideje kako riješiti zadatke. U suprotnom, rad na vježbi neće biti produktivan. Zadatke koje studenti ne stignu uraditi za vrijeme vježbe, trebali bi ih samostalno uraditi kod kuće.

1. Definirajte i implementirajte klasu "`Vektor3d`" u skladu sa deklaracijom i implementacijom koja je prikazana na Predavanju 9_b (treba uključiti sve što je za tu klasu razvijeno, uključujući i prijateljsku funkciju "`ZbirVektora`"), s tim što ćete dodati još tri nove metode "`PostaviX`", "`PostaviY`" i "`PostaviZ`" koje treba da omoguće neovisnu izmjenu pojedinačnih koordinata vektora. Napišite i mali testni program u kojem ćete demonstrirati *sve elemente* iz interfejsa razvijene klase.

2. Izmijenite implementaciju klase razvijene u prethodnom zadatku tako da se za čuvanje koordinata vektora umjesto tri privatna atributa "`x`", "`y`" i "`z`" koji su tipa realnih brojeva koristi jedan privatni atribut nazvan "`koordinate`" koji je tipa niza od tri realna elementa. Izmjenu treba izvesti tako da zaglavlja svih metoda unutar interfejsa klase ostanu neizmijenjena. Demonstrirajte da će testni program napisan u prethodnom zadatku raditi bez ikakvih izmjena sa ovako izmijenjenom klasom. Ovo zapravo ilustrira osnovnu poentu razdvajanja interfejsa od implementacije: *projektant klase može u potpunosti izmijeniti način kako je interno organizirana klasa a da korisnik klase to uopće ne primijeti, sve dok interfejs ostaje isti.*

3. Proširite klasu iz prethodnog zadatka tako što ćete dodati novu metodu nazvanu "`DajBrojIspisa`" koja daje kao rezultat broj koliko puta je do trenutka poziva metode objekat (vektor) nad kojim je primijenjena ispisan na ekran (pozivom metode "`Ispisi`"). Na primjer, ako se izvrše naredbe

   ```
   Vektor3d v1, v2;
   v1.Postavi(3, 4, 2); v2.Postavi(2, 0, 5);
   v1.Ispisi(); v1.Ispisi(); v1.Ispisi(); v2.Ispisi(); v2.Ispisi();
   std::cout << std::endl;
   std::cout << "Objekat v1 je ispisan " << v1.DajBrojIspisa() << "puta , a objekat v2 "
     << v2.DajBrojIspisa() << " puta" << std::endl;
   ```

   posljednja naredba treba da proizvede ispis "Objekat v1 je ispisan 3 puta, a objekat v2 2 puta". Implementaciju klase možete mijenjati sa ciljem da ovo postignete, ali ništa u interfejsu ne smijete izmijeniti (osim dodavanja nove metode). Posebno, sve metode koje su inspektori moraju i dalje ostati inspektori.

4. Definirajte i implementirajte klasu "`Sat`" koja predstavlja digitalni sat. Implementacija klase treba se zasnivati na tri cjelobrojna privatna atributa koji čuvaju trenutni broj sati, minuta i sekundi. Interfejs klase treba da sadrži sljedeće elemente. Na prvom mjestu, tu je statička funkcija članica (metoda) "`DaLiJeIspravno`", koja prima tri cjelobrojna parametra koji respektivno predstavljaju broj sati minuta i sekundi, a koja vraća logičku vrijednost "tačno" ili "netačno", ovisno od toga da li ti parametri mogu predstavljati ispravno vrijeme ili ne (sati moraju biti u opsegu od 0 do 23 a minute i sekunde u opsegu od 0 do 59 uključivo). Metoda "`Postavi`" također prima tri cjelobrojna parametra, a vrši postavljanje sata na iznos sati, minuta i sekundi koji je zadan putem parametara. Ova metoda baca izuzetak tipa "`domain_error`" uz prateći tekst "Neispravno vrijeme" ukoliko se proslijede neispravni parametri. Slična je i metoda "`PostaviNormalizirano`", samo što ona nikada ne baca izuzetak, nego vrši "normalizaciju" eventualno neispravno zadanog vremena (prelijevom viška sekundi u minute, viška minuta u sate i viška sati u novi dan). Recimo, ukoliko se zada 25 sati, 150 minuta i 290 sekundi, to treba normalizirati u 3 sata, 34 minute i 50 sekundi. Normalizacija treba da radi i za negativne vrijednosti parametara, tako da se 0 sati, 0 minuta i -1 sekunda normalizira u 23 sata, 59 minuta i 59 sekundi. Metoda "`Sljedeci`" bez parametara treba da poveća vrijeme zapamćeno u satu za 1 sekundu (npr. ukoliko je tekuće vrijeme "12:48:59", nakon poziva ove metode vrijeme treba da postane "12:49:00"). Slično, metoda "`Prethodni`" (koja također nema parametara) treba da smanji vrijeme zapamćeno u satu za 1 sekundu, dok metoda "`PomjeriZa`" predstavlja generalizaciju prethodne dvije metode tako što vrši pomak tekućeg vremena za broj sekundi koji je zadan putem cjelobrojnog parametra (pomjeranje je unazad ukoliko se proslijedi negativna vrijednost kao parametar). Sve ove tri metode također vraćaju kao rezultat izmijenjeni objekat, sa ciljem da se omogući kaskadno pozivanje poput "`s.Sljedeci().Sljedeci()`". Metoda "`Ispisi`" ispisuje stanje sata u obliku "$hh:mm:ss$". Metode "`DajSate`", "`DajMinute`" i "`DajSekunde`"

vraćaju kao rezultat trenutni broj sati, minuta i sekundi u tekućem vremenu. Sve ove metode su bez parametara. Sve metode koje su po svojoj prirodi inspektori obavezno treba deklarirati kao takve (da bi se omogućio njihov poziv nad konstantnim objektima). Pored navedenih elemenata, potrebno je još realizirati i prijateljsku funkciju nazvanu "`BrojSekundiIzmedju`", kao i statičku metodu (tj. statičku funkciju članicu) nazvanu "`Razmak`". Obje ove funkcije rade potpuno istu stvar, a funkcionalnost je duplirana čisto sa ciljem da se upoznate sa dva različita pristupa istom problemu. Ove funkcije primaju kao parametre dva objekta tipa "`Sat`", a vraćaju kao rezultat broj sekundi između vremena zapisanih u prvom i drugom parametru (rezultat je pozitivan ukoliko je vrijeme u prvom parametru veće nego vrijeme u drugom parametru, a negativan u suprotnom), Obavezno napišite i testni program u kojem će se upotrebiti *svi elementi* interfejsa napisane klase.

5. Definirajte i implementirajte klasu "`Sat`" koja ima potpuno isti interfejs i potpuno isto ponašanje kao klasa iz prethodnog zadatka, samo čija se interna struktura umjesto tri atributa koja čuvaju trenutni broj sati, minuta i sekundi sastoji samo od jednog atributa, koji čuva *ukupan broj sekundi* (npr. umjesto informacije "3 sata, 20 minuta, 15 sekundi" čuva se samo informacija koja kaže 12015 sekundi). Mada će ovo možda tražiti izmjenu implementacije *svih* (ili *skoro svih*) *metoda klase* (pri tipičnim izvedbama, metoda "`Postavi`" će se sasvim neznatno izmijeniti, metode "`Sljedeci`", "`Prethodni`" i "`PomjeriZa`" kao i prijateljska funkcija "`BrojSekundiIzmedju`" odnosno statička metoda "`Razmak`" bitno će se pojednostaviti, metode "`DajSate`", "`DajMinute`", "`DajSekunde`" će se zakomplicirati, dok će se metoda "`Ispisi`" možda zakomplicirati, a možda će i ostati potpuno ista, ako ste je pametno realizirali), pokažite da će testni program iz prethodnog zadatka bez ikakve prepravke raditi sa ovako modificiranom klasom. Savjet: da biste vršili što manje prepravki, probajte se prilikom rješavanja zadatka 4 u realizaciji funkcija članica što je god moguće više oslanjati na druge funkcije članice, a što manje na to šta su zaista atributi klase.