Zadaci za Tutorijal 10.

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

1. Definirajte i implementirajte klasu "`StedniRacun`". U klasi treba da postoji konstruktor koji postavlja stanje računa za zadanu početnu vrijednost (realni broj), ili na nulu ukoliko se nikakva početna vrijednost ne zada. U slučaju da je početna vrijednost negativna, konstruktor treba baciti izuzetak tipa "`logic_error`" uz prateći tekst "Nedozvoljeno pocetno stanje". Metode "`Ulozi`" i "`Podigni`" imaju parametar koji je realni broj, a simuliraju ulaganje zadanog iznosa odnosno podizanje zadanog iznosa sa računa. Pri tome, podizanje je dozvoljeno jedino ukoliko je iznos koji se podiže manji ili jednak od trenutnog stanja računa, u suprotnom se baca izuzetak tipa "`logic_error`" uz prateći tekst "Transakcija odbijena". Metoda "`DajStanje`" nema parametara, a daje kao rezultat trenutno stanje računa. Konačno, metoda "`ObracunajKamatu`" ima realni parametar koji predstavlja kamatnu stopu *u procentima*, a uvećava glavnicu (trenutno stanje) za iznos kamate koja se dobija množenjem glavnice sa kamatnom stopom. Kamatna stopa mora biti pozitivna, u suprotnom se baca izuzetak tipa "`logic_error`" uz prateći tekst "Nedozvoljena kamatna stopa". Sve metode koje su inspektori obavezno deklarirajte kao takve. Obavezno napišite i testni program u kojem ćete testirati sve elemente razvijene klase (naročito trebate testirati mogu li se inspektori korektno pozvati nad konstantnim objektima tipa "`StedniRacun`").

2. Definirajte i implementirajte klasu "`Krug`" koja predstavlja krug, te klasu "`Valjak`" koja se oslanja na klasu "`Krug`". Jedini atribut klase "`Krug`" je poluprečnik kruga (tipa "**`double`**"). Konstruktor ove klase omogućava da se poluprečnik zada putem parametra, pri čemu nije moguće kreirati objekte tipa "`Krug`" a da se parametar ne zada. Također, ne smije biti dozvoljena automatska pretvorba realnih brojeva u objekte tipa "`Krug`". Treba podržati i metodu "`Postavi`" koja radi istu stvar kao konstruktor, samo što omogućava naknadnu izmjenu poluprečnika već kreiranog objekta. U oba slučaja, ukoliko je poluprečnik negativan ili nula, treba baciti izutetak tipa "`domain_error`" uz prateći tekst "Neispravan poluprecnik". Metode "`DajPoluprecnik`", "`DajObim`" i "`DajPovrsinu`" bez parametara daju respektivno kao rezultat poluprečnik, obim i površinu kruga nad kojim su pozvani (uzmite da je $\pi = 4 \cdot \arctan 1$). Metoda "`Skaliraj`" prima kao parametar realni broj (faktor skaliranja) i množi poluprečnik sa faktorom skaliranja. Izuzetak tipa "`domain_error`" uz prateći tekst "Neispravan faktor skaliranja" baca se ukoliko je faktor skaliranja negativan ili nula. Konačno, metoda "`Ispisi`" nema parametara, a ispisuje podatke o krugu u formatu "`R=... O=... P=...`" gdje se umjesto tri tačke nalaze informacije o poluprečniku, obimu i površini respektivno. Klasa "`Valjak`" treba da ima dva atributa; "`baza`" tipa "`Krug`" i "`visina`" tipa "**`double`**". Konstruktor klase "`Valjak`" zahtijeva dva parametra koji redom predstavljaju poluprečnik baze i visinu valjka. Primjerci ove klase također se ne mogu kreirati bez zadavanja pomenutih informacija. Za nju također treba podržati i funkciju "`Postavi`" koja radi istu stvar kao konstruktor, samo što omogućava naknadnu izmjenu već kreiranog objekta. U oba slučaja, ukoliko su poluprečnik baze ili visina negativni, treba baciti izuzetak tipa "`domain_error`" uz prateći tekst "Neispravan poluprecnik" ili "Neispravna visina" (u slučaju da su oba parametra negativna, prednost treba dati prvom tekstu). Metode "`DajBazu`", "`DajPoluprecnikBaze`", "`DajVisinu`", "`DajPovrsinu`" i "`DajZapreminu`" nemaju parametara a redom vraćaju kao rezultat bazu (kao objekat tipa "`Krug`"), poluprečnik baze, visinu, površinu odnosno zapreminu valjka (valjda znate da je površina valjka dvostruka površina baze plus obim baze pomnožen sa visinom, dok je zapremina valjka površina baze pomnožena sa visinom). I za ovu klasu je podržana "`Skaliraj`" koja kao parametar prima faktor skaliranja i množi poluprečnik baze te visinu sa faktorom skaliranja, uz bacanje izuzetka tipa "`domain_error`" uz prateći tekst "Neispravan faktor skaliranja" ukoliko je faktor skaliranja negativan ili nula. Konačno, metoda bez parametara "`Ispisi`" ispisuje podatke o valjku u formatu "`R=... H=... P=... V=...`", gdje umjesto tri tačke treba respektivno smjestiti informacije o poluprečniku baze, visini, površini i zapremini valjka. Sve metode koje su inspektori obavezno deklarirajte kao takve. Obavezno napišite i testni program u kojem ćete testirati sve elemente razvijenih klasa (naročito trebate testirati mogu li se inspektori korektno pozvati nad konstantnim objektima tipa "`Krug`" i "`Valjak`").

3. Klase "Vektor3d" iz zadataka 1, 2 i 3 sa prethodnog tutorijala, imaju nekoliko nedostataka. Na prvom mjestu, objekti tipa "Vektor3d" rađaju se sa nasumičnim početnim sadržajem, Proširite klasu iz Zadatka 3. sa prethodnog tutorijala konstruktorom bez parametara koji kreira vektor čije su sve tri koordinate postavljene na 0, kao i konstruktorom sa tri parametra koji obavlja istu funkciju kao i "Postavi" sa tri parametra, samo odmah pri kreiranju objekta. Dalje, brojanje ispisa je imalo "bag" (koji tada nismo znali ispraviti) što pri kopiranju objekta, novokreirana kopija nasljeđuje brojač ispisa od svog originala. Recimo, ako se izvrši nešto poput

```
Vektor3d v1; v1.Postavi(1,2,3); v1.Ispisi(); v1.Ispisi(); v1.Ispisi()
Vektor3d v2(v1); v2.Ispisi(); v2.Ispisi();
```

tada će poziv "v2.DajBrojIspisa()" dati vrijednost 5, iako je objekat "v2" ispisan samo dva puta. Izvršite korekcije koje su potrebne da se ispravi ovaj bag (odgovor kako to uraditi krije se u elementima koji su uvedeni na Predavanju 10).

4. Definirajte i implementirajte klasu "Ugao" (ili "Kut", u skladu sa Vašim jezičkim opredjeljenjem) koja omogućava rad sa uglovima (kutovima) u ravni. Klasa treba da ima sljedeći interfejs:

```
Ugao(double radijani = 0);
Ugao(int stepeni, int minute, int sekunde);
void Postavi(double radijani);
void Postavi(int stepeni, int minute, int sekunde);
double DajRadijane() const;
void OcitajKlasicneJedinice(int &stepeni, int &minute, int &sekunde);
int DajStepene() const;
int DajMinute() const;
int DajSekunde() const;
void Ispisi() const;
void IspisiKlasicno() const;
Ugao &SaberiSa(const Ugao &u);
Ugao &PomnoziSa(double x);
friend Ugao ZbirUglova(const Ugao &u1, const Ugao &u2);
friend Ugao ProduktUglaSaBrojem(const Ugao &u, double x);
```

Konstruktor sa jednim parametrom postavlja vrijednost ugla u radijanima (ovaj parametar ima podrazumijevanu vrijednost 0 što omogućava da se ovaj konstruktor koristi i kao konstruktor bez parametara, pri čemu se kreira prazan ugao od 0 radijana), dok konstruktor sa tri parametra postavlja vrijednost ugla u stepenima, minutama i sekundama. Pri tome se svi uglovi reduciraju na opseg od 0 do 2π odnosno od 0 do $360°$ tako da se, na primjer, ugao od $5\pi/2$ odnosno $450°$ automatski reducira na vrijednost $\pi/2$ odnosno $90°$, dok se ugao od $-\pi/4$ odnosno $-45°$ automatski reducira na vrijednost $315°$ odnosno $7\pi/4$. Konstruktor sa jednim parametrom treba podržavati automatsku konverziju realnih brojeva u objekte tipa "Ugao" (ili "Kut"). Dvije metode "Postavi" (sa jednim i tri parametra) obavljaju isti posao kao i konstruktori sa jednim odnosno tri parametra respektivno, a služe za naknadnu promjenu ugla. Metoda "DajRadijane" vraća vrijednost ugla u radijanima. Metoda "OcitajKlasicneJedinice" očitava vrijednost ugla u stepenima, minutama i sekundama i smješta očitane vrijednosti u odgovarajuće parametre metode. Metode "DajStepene", "DajMinute" i "DajSekunde" omogućavaju da se istim ovim informacijama pristupi neovisno, a ne isključivo "u paketu". Metoda "Ispisi" ispisuje vrijednost ugla u radijanima, dok metoda "IspisiKlasicno" ispisuje vrijednost ugla u stepenima, minutama i sekundama u obliku poput "23deg 8min 47sec". Metoda "SaberiSa" dodaje ugao zadan parametrom na ugao nad kojim je primijenjena i usput vraća kao rezultat tako modificiran ugao. Metoda "PomnoziSa" množi ugao nad kojim je primijenjena sa brojem koji je zadan parametrom i usput vraća kao rezultat tako modificiran ugao. Obje ove metode trebaju obezbijediti da nakon izvršene operacije rezultat bude reduciran na opseg $0-2\pi$ ($0-360°$). Konačno, dvije prijateljske funkcije "ZbirUglova" i "ProduktUglaSaBrojem" vraćaju kao rezultat novi ugao koji je jednak zbiru uglova zadanih parametrima, odnosno produktu ugla i broja koji su zadani putem parametara. Implementaciju klase treba zasnovati na jednom privatnom atributu koji čuva vrijednost ugla u radijanima (realan broj). Napišite i kratki testni program u kojem ćete demonstrirati da svi elementi napisane klase rade u skladu sa specifikacijama. Posebno treba provjeriti da li radi sabiranje objekata tipa "Ugao" (ili "Kut") sa realnim brojem.

5. Izmijenite implementaciju klase razvijene u prethodnom zadatku tako da se informacija o uglu umjesto u radijanima interno čuva u tri cjelobrojna atributa koji redom sadrže broj stepeni, minuta i sekundi koje čine ugao. Iako će ova izmjena možda tražiti promjenu implementacije gotovo svih metoda klase (ovisno o tome kako ste ih pisali), uvjerite se da će nakon te izmjene testni program koji koristi napisanu klasu i dalje raditi posve identično, bez ikakvih izmjena.