POPRAVNI ISPIT IZ PREDMETA “TEHNIKE PROGRAMIRANJA” (I PARCIJALNI)

Zadatak 1. (5 poena)

Utvrdite šta će ispisati sljedeći program (odgovor treba biti obrazložen):

```
#include <iostream>
#include <complex>
using namespace std;
double p(double x, int y = 0) {
  return 2 * x + 3 * y;
}
int q(int &y) {
  y--; return y * 2;
}
double p(int x) {
  return 3 * x;
}
int r(int &z) {
  ++z; return z - 1;
}
void s(int &x, int y, int z(int &x)) {
  x += z(y);
  cout << complex<double>(x++, y) << endl;
}
int main() {
  int (*t[3])(int &x) = {q, r, q};
  int y(3), z(4);
  cout << p(2) << "," << p(2.) << "," << p(2, 2) << endl;
  for(int x = 0; x < y; x++) s(z, x, t[x]);
  cout << z << endl;
  return 0;
}
```

Zadatak 2. (2,5 poena)

Za neki broj kažemo da je *savršen* ukoliko je jednak sumi svih svojih djelilaca. Na primjer, 28 je savršen broj: njegovi djelioci su 1, 2, 4, 7 i 14, a 1 + 2 + 4 + 7 + 14 = 28. Napišite funkciju koja prima vektor cijelih brojeva kao parametar, a koja kao rezultat vraća novi vetor koji sadrži samo one brojeve iz vektora koji je zadan kao parametar koji su savršeni brojevi. Napišite i kratki isječak programa koji će tražiti da se sa tastature unese slijed cijelih brojeva pri čemu 0 označava kraj unosa i koji će na kraju ispisati koji su među unesenim brojevima bili savršeni brojevi.

Zadatak 3. (2,5 poena)

Napišite funkciju “`OdstraniParneCifre`” koja ima jedan cjelobrojni parametar. Funkcija treba da transformira taj parametar, tako da se kao rezultat transformacije dobije novi broj iz kojeg su odstranjene sve cifre koji su parni brojevi, dok ostale cifre zadržavaju isti poredak kao u izvornom broju. Na primjer, ako se funkciji proslijedi promjenljiva koja sadrži vrijednost 32564718, ta ista promjenljiva po završetku funkcije imaće vrijednost 3571. Ukoliko su sve cifre bile parne, kao rezultat transformacije se dobija 0. Napišite i kratki isječak programa u kojem ćete demonstrirati kako se upotrebljava napisana funkcija.

Zadatak 4. (2,5 poena)

Nazovimo neku riječ “korektnom” ukoliko se u njoj ne pojavljuje skupina uzastopnih suglasnika duža od 2 slova. Na primjer, riječ “badlekuvje” je korektna, a riječ “balgftekrihvnu” nije korektna u smislu date definicije, jer sadrži 2 grupe suglasnika sa više od 2 uzastopna suglasnika (“lgft” i “hvn”). Napišite funkciju koja prihvata string kao parametar, a koja kao rezultat vraća logičku vrijednost “tačno” ukoliko string koji joj je proslijeđen kao parametar predstavlja korektnu riječ u smislu gore date definicije, a u suprotnom vraća logičku vrijednost “netačno”.

Zadatak 5. (2,5 poena)

Napišite generičku funkciju “`Analiza`” sa 4 parametra. Prva dva parametra omeđuju blok elemenata za koje se pretpostavlja da se mogu međusobno porediti (tj. prvi parametar pokazuje na početak bloka, a drugi parametar neposredno iza njegovog kraja). Funkcija treba da pretraži zadani blok elemenata i da *smjesti* u treći i četvrti parametar respektivno broj elemenata u bloku koji su jednaki najmanjem, odnosno najvećem elementu bloka. Napišite i isječak programa u kojem ćete demonstrirati kako se napisana funkcija može iskoristiti na primjeru jednog fiksnog niza od 10 realnih brojeva.

Zadatak 6. (2,5 poena)

Postoje razni metodi pomoću kojih je moguće izračunati približnu vrijednost integrala neke funkcije $f(x)$ na intervalu (a, b). Jedan od najjednostavnijih ali ujedno i najmanje tačnih metoda je tzv. *trapezno ili Eulerovo pravilo*, prema kojem je

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{b-a}{n}\left[\frac{1}{2}f(a)+\frac{1}{2}f(b)+\sum_{k=1}^{n-1} f\left(a+\frac{b-a}{n}k\right)\right]$$

gdje je n broj podintervala na koji dijelimo interval (a, b). Veći broj podintervala daje i veću tačnost, ali do određene granice. Napišite funkciju “`TrapeznoPravilo`” koja prima kao parametre f, a, b i n (f je *funkcija čiji se integral računa*) a koja kao rezultat daje približnu vrijednost integrala računatu pomoću trapeznog pravila. Parametar n treba imati podrazumijevanu vrijednost 100, koja se koristi ukoliko se prilikom poziva funkcije ovaj parametar izostavi. Napišite i isječak programa u kojem ćete demonstrirati kako se napisana funkcija može iskoristiti da nađete približnu vrijednost integrala funkcije $1/x$ na intervalu $(1, 2)$.

Zadatak 7. (2,5 poena)

Napišite funkciju sa jednim parametrom n koja kreira dinamički niz od n cijelih brojeva, popunjava ga sa prvih n Fibonačijevih brojeva i vraća kao rezultat pokazivač na prvi element kreiranog niza (Fibonačijevi brojevi F_n definirani su relacijom $F_n = F_{n-1} + F_{n-2}$, pri čemu je $F_1 = F_2 = 1$). Ukoliko parametar n nije prirodan broj, ili ukoliko nema dovoljno memorije za alokaciju, funkcija treba baciti izuzetak. Napišite i isječak programa u kojem ćete sa tastature unijeti broj n, kreirati dinamički niz prvih n Fibonačijevih brojeva putem napisane funcije, ispisati njegove elemente na ekran i, na kraju, izbrisati alocirani niz. U tom isječku obavezno predvidite i hvatanje izuzetaka koji mogu eventualno biti bačeni iz funkcije.

POPRAVNI ISPIT IZ PREDMETA "TEHNIKE PROGRAMIRANJA" (II PARCIJALNI)

NAPOMENA: U svim klasama koje treba razviti, trivijalne metode koje se mogu implementirati u jednoj ili najviše dvije naredbe možete implementirati direktno unutar deklaracije klase, dok sve ostale metode trebate implementirati izvan deklaracije klase.

Zadatak 1 (7 poena)

Definirajte i implementirajte klasu "`Polinom`", koja omogućava rad sa polinomima. Klasa treba da ima konstruktor sa jednim parametrom koji predstavlja stepen polinoma. Ovaj konstruktor ne smije se koristiti za automatsku pretvorbu cijelih brojeva u objekte tipa "`Polinom`". Koeficijenti polinoma treba da se čuvaju u vektoru realnih brojeva (tj. atributu tipa "`vector<double>`"). Klasa bi trebala da ima preklopljeni operator "`[]`" kojim se omogućava pristup koeficijentima polinoma. Indeksi koeficijenata polinoma se kreću od nule (slobodni član) do stepena polinoma, a u slučaju indeksa izvan opsega treba baciti izuzetak. Za računanje vrijednosti polinoma koristi se preklopljeni operator "`()`" kojem se prosljeđuje vrijednost argumenta za koji treba izračunati vrijednost polinoma. Treba podržati i binarne operatore "`+`", "`-`" i "`*`" za sabiranje, oduzimanje i množenje dva polinoma, kao i za sabiranje oduzimanje i množenje polinoma *sa realnim brojem* (sabiranje i oduzimanje polinoma sa realnim brojem utiče samo na slobodni član, dok se u slučaju množenja polinoma sa brojem svi koeficijenti množe sa zadanim brojem). Potrebno je podržati i bliske srodnike ovih operatora "`+=`", "`-=`" i "`*=`", kao i unarni operator "`-`" koji kao rezultat daje polinom sa izvrnutim predznacima svih koeficijenata. Dalje, treba podržati i binarni relacioni operator "`==`" i "`!=`" koji testiraju jednakost, odnosno nejednakost dva polinoma. Konačno, treba podržati i operator za ispis "`<<`" koji ispisuje polinom na ekran, koristeći "x" kao ime nezavisne varijable. Radi jednostavnosti, ispisujte koeficijent čak i u slučaju kad je on jednak jedinici, ali u slučaju da je koeficijent jednak nuli, odgovarajući član ne treba ispisivati. Na primjer, uz pretpostavku da su koeficijenti polinoma redom 3, 0, 2, –4, 0, –1 i 6.ispis treba izgledati ovako:

```
3 + 2 x^2 - 4 x^3 - 1 x^5 + 6 x^6
```

Zadatak 2 (13 poena)

Za potrebe vođenja evidencije o knjigama u nekoj biblioteci, u evidencijskom programu koriste se klase nazvane "`Knjiga`", "`Udzbenik`" i "`Biblioteka`". Vaš zadatak je da definirate i implementirate te klase i da napišete glavni program koji koristi te klase.

Klasa "`Knjiga`" opisuje podatke o jednoj knjizi, a njeni atributi su evidencijski broj (tipa cijeli broj), puni naslov koji uključuje ime pisca i ime djela (tipa string, sve u jednom atributu), članski broj čitaoca kod koga se trenutno nalazi knjiga ili 0 ukoliko knjiga trenutno nije na čitanju tj. ukoliko je trenutno raspoloživa (tipa cijeli broj), te podatak koliko dugo dana je knjiga na čitanju (ukoliko knjiga trenutno nije na čitanju, vrijednost ovog podatka je nebitna). Konstruktor klase inicijalizira podatke o evidencijskom broju i naslovu knjige na vrijednosti zadane parametrima. Pri tome se podaci o čitaocu inicijaliziraju tako da signaliziraju da je knjiga slobodna. Vrijednost svih privatnih podataka može se saznati pomoću odgovarajućih privatnih metoda. Klasa posjeduje i metodu koja knjigu proglašava "zaduženom", pri čemu je njen parametar članski broj čitaoca koji zadužuje knjigu, te metodu bez parametara koja "razdužuje" knjigu. Pri zaduženju, informacija o tome koliko dugo je knjiga na čitanju postavlja se na nulu. Postoji i metoda koja daje informaciju da li je knjiga zadužena ili nije. Podržana je i virtuelna metoda koja ispisuje na izlazni tok podatke o evidencijskom broju i naslovu knjige. Parametar metode je referenca na objekat toka preko kojeg se vrši ispis (tipa "`ostream`"). Konačno, klasa "`Knjiga`" posjeduje i virtuelnu metodu koja vraća informaciju da li je knjiga udžbenik ili ne (ova metoda će se koristiti u konstruktoru kopije klase "`Biblioteka`"). U klasi "`Knjiga`" ova metoda će uvijek vraćati logičku vrijednost "netačno" (dok će odgovarajuća verzija ove metode u klasi "`Udzbenik`" vraćati logičku vrijednost "tačno").

Klasa "`Udzbenik`" je naslijeđena iz klase "`Knjiga`". U odnosu na baznu klasu "`Knjiga`", ova klasa sadrži i dodatni atribut koji predstavlja predmet za koji je udžbenik namijenjen (tipa string) te

odgovarajuću pristupnu metodu pomoću koje se može saznati za koji je predmet udžbenik namijenjen. Konstruktor ove klase je proširen da omogući i zadavanje predmeta za koji je udžbenik namijenjen. Privatna metoda za ispis je izmijenjena da predvidi i ispis predmeta za koji je udžbenik namijenjen, dok privatna metoda koja govori da li je knjiga udžbenik ili ne u ovoj klasi uvijek vraća logičku vrijednost "tačno". Svi ostali elementi klase "`Udzbenik`" se prosto preuzimaju iz klase "`Knjiga`".

Klasa "`Biblioteka`" opisuje kolekciju knjiga (uključujući i udžbenike). Ova klasa sadrži informacije o broju knjiga u biblioteci, kao i o maksimalnom broju knjiga koje biblioteka može evidentirati (ova informacija treba biti izvedena kao konstantni atribut). Podaci o evidentiranim knjigama čuvaju se u dinamički alociranim objektima tipa "`Knjiga`" ili "`Udzbenik`" kojima se pristupa preko dinamički alociranog niza pokazivača na objekte tipa "`Knjiga`", a na koji opet pokazuje privatni dvojni pokazivač. Konstruktor klase vrši dinamičku alokaciju memorije, pri čemu njegov parametar predstavlja maksimalan broj knjiga koje biblioteka može evidentirati (ovaj konstruktor se ne smije koristiti za automatsku konverziju cijelih brojeva u objekte tipa "`Biblioteka`"). Podržan je odgovarajući destruktor koji se brine za oslobađanje zauzete memorije, konstruktor kopije koji obezbjeđuje pouzdano kopiranje objekata tipa "`Biblioteka`" zasnovano na strategiji dubokog kopiranja (s obzirom da se radi o polimorfnoj kolekciji, iskoristite metodu koja govori da li je knjiga udžbenik ili ne da biste znali koji objekat da kreirate prilikom kopiranja), kao i operator dodjele koji omogućava međusobno dodjeljivanje, ali samo između biblioteka koje imaju *iste kapacitete* (u suprotnom, treba baciti izuzetak).

Od metoda, na prvom mjestu tu su metode koje kreiraju novu knjigu (običnu) odnosno novi udžbenik i evidentiraju ih u biblioteci. Parametri ovih metoda su isti kao u konstruktoru klasa "`Knjiga`" odnosno "`Udzbenik`". Sljedeća metoda je metoda koja vraća referencu na knjigu čiji se evidencijski broj navodi kao parametar (ova metoda će biti od velike koristi za realizaciju većine ostalih metoda). U slučaju da ne postoji knjiga sa zadanim evidencijskim brojem treba baciti izuzetak (to vrijedi i za sve ostale metode u kojima se traže knjige sa zadanim evidencijskim brojem). Dalje, postoje metode koje "zadužuju" odnosno "razdužuju" knjigu. Njihovi parametri su evidencijski broj knjige, te u slučaju zaduživanja, članski broj čitaoca koji zadužuje knjigu. Tu je i metoda koja vraća članski broj čitaoca kod kojeg se nalazi knjiga čiji je evidencijski broj zadan kao parametar, odnosno nulu ukoliko knjiga nije zadužena (sve ove metode najlakše je realizirati pozivom metode koja daje referencu na knjigu sa zadanim evidencijskim brojem). Predviđene su i metode koje ispisuju podatke o svim slobodnim knjigama, svim zaduženim knjigama, te knjigama koje su na čitanju duže od *n* dana, pri čemu se *n* zadaje kao parametar. Konačno, postoji i metoda koja sortira sve knjige u takav poredak da se knjige koje se su duže vremena zadužene nađu ispred onih koje su kraće vremena zadužene. Ukoliko dvije knjige imaju isto trajanje zaduženja, knjiga čiji naziv dolazi prije po abecedi treba da bude ispred knjige koja dolazi kasnije po abecedi.

Napišite i glavni program, koji će iz tekstualnih datoteka pročitati podatke o knjigama i njihovim zaduženjima i nakon toga ispisati podatke o svim zaduženim i svim slobodnim knjigama. Podaci o knjigama nalaze se u tekstualnoj datoteci "`KNJIGE.TXT`". Ova datoteka je organizirana tako da se za svaku knjigu u po jednom redu prvo nalazi evidencijski broj knjige, a zatim i njen naziv (između se nalazi razmak). Ukoliko je u pitanju udžbenik, ispred evidencijskog broja nalazi se slovo "`U`", a u sljedećem redu se nalazi i podatak o predmetu za koji je udžbenik namijenjen. Na primjer, datoteka "`KNJIGE.TXT`" može izgledati ovako:

```
1234 Mehmed Meša Selimović: Derviš i smrt
U4312 Momčilo Ušćumlić: Zbirka zadataka iz više matematike
Inžinjerska matematika I/II
3124 Ivo Andrić: Prokleta avlija
...
```

Podaci o zaduženjima nalaze se u datoteci "`ZADUZENJA.TXT`". Svaki red ove datoteke predstavlja po jednu zaduženu knjigu, a u njemu se nalaze evidencijski broj knjige i članski broj čitaoca koji je zadužio knjigu, razdvojeni zarezom (npr. "`4312,344`"). Radi jednostavnosti, možete pretpostaviti da obje datoteke sigurno sadrže ispravne podatke.