Zadaci za Laboratorijsku vježbu 4

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. Napišite funkciju "`Cifre`" koja treba da ima 3 parametra "`n`", "`c_min`" i "`c_max`", pri čemu parametar "`n`" predstavlja neki cijeli broj, a tip mu je "`long long int`". Funkcija treba da pronađe najmanju i najveću cifru u parametru "`n`" i da smjesti pronađene cifre u parametre "`c_min`" i "`c_max`" (ovi parametri trebaju biti tipa "`int`", jer su vrijednosti cifri svakako ograničene na opseg od 0 do 9). Pored toga, funkcija treba da kao rezultat vrati broj cifara u parametru "`n`". Na primjer, naredbe

   ```
   Cifre(372324412645, a, b);
   e = Cifre(-54746639, c, d);
   ```

   treba da u promjenljive "`a`" i "`b`" odnosno "`c`" i "`d`" (pod uvjetom da su propisno deklarirane) smjesti brojeve 1 i 7, odnosno 3 i 9, jer su upravo to najmanja i najveća cifra u broju 372324412645 odnosno broju −45746639. Pored toga, u promjenljivu "`e`" smjestiće se broj 8 (broj cifara u broju −45746639). Funkcija ne smije koristiti nikakve pomoćne nizove ili vektore (samo individualne promjenljive), kao ni nepredznačne tipove podataka. Napišite i kratki testni program u kojem ćete demonstrirati napisanu funkciju.

2. Napišite funkciju "`IzvrniString`" sa jednim parametrom tipa "`string`". Funkcija treba da *ispremješta* elemente stringa tako da prvi znak postane posljednji, a posljednji prvi. Na primjer, ukoliko se u glavnom programu izvrši sekvenca naredbi

   ```
   std::string s = "Ovo je neki tekst...";
   Izvrni(s);
   std::cout << s;
   ```

   ispis na ekranu treba da bude "`...tsket iken ej ovO`". Funkcija treba da bude realizirana tako da vrši premještanje "u mjestu", odnosno da ne koristi *nikakav pomoćni string* osim stringa koji je prenesen kao parametar (niti pomoćni niz, vektor ili nešto slično). Funkcija ne vraća nikakvu vrijednost nego samo modificira string koji joj je poslan kao parametar. Napišite i kratki glavni program u kojoj ćete demonstrirati napisanu funkciju na rečenici koja se unosi sa tastature.

3. Napišite funkciju "`IzdvojiKrajnjeRijeci`" sa 3 parametra, čiji je prvi parametar vektor stringova (tj. vektor čiji su elementi tipa "`string`") za koji se pretpostavlja da sadrži neki spisak riječi (svaka riječ je u posebnom stringu). Funkcija treba da pronađe prvu i posljednju riječ po abecednom poretku (ne praveći pri tome razliku između malih i velikih slova) i da ih smjesti redom u drugi i treći parametar funkcije. U slučaju praznog vektora, u drugi i treći parametar treba smjestiti prazan string. Zatim napišite funkciju "`ZadrziDuplikate`" čiji je jedini parametar ponovo vektor stringova za koji se pretpostavlja da sadrži neki spisak riječi. Funkcija treba da modificira ovaj vektor stringova, tako što će u njemu zadržati samo one stringove (riječi) koji su duplikati, tj. koji se u vektoru pojavljuju više od jedanput, pri čemu svaki duplikat treba zadržati samo jedanput (tačnije, prilikom njegovog prvog pojavljivanja). Recimo, ukoliko vektor sadrži redom riječi "xyzzy", "qwert", "uiop", "asd", "rrrfat", "asd", "yxcvbb", "qwert", "asd" i "cvbnm", nakon poziva ove funkcije isti vektor treba da sadrži samo riječi "qwert" i "asd" (u tom poretku), jer su ovo jedine dvije riječi koje se pojavljuju više od jedanput unutar vektora. Za realizaciju ove funkcije *nije dozvoljeno koristiti funkciju* "`erase`". Napisane funkcije iskoristite u testnom programu koji traži od korisnika da unese spisak riječi (broj riječi se prethodno unosi sa tastature), a zatim ispisuje na ekran prvu i poslednju riječ iz spiska po abecednom poretku, kao i popis svih riječi koje se ponavljaju. Dijalog između korisnika i programa bi trebao izgledati poput sljedećeg:

   ```
   Koliko zelite unijeti rijeci: 10
   Unesite rijeci: xyzzy qwert uiop asd rrrfat asd yxcvbb qwert asd cvbnm
   Prva rijec po abecednom poretku je: asd
   Posljednja rijec po abecednom poretku je: yxcvbb
   Rijeci koje se ponavljaju su: qwert asd
   ```

   U obje napisane funkcije nije dozvoljeno sortirati sadržaj vektora.

4. Napišite generičku funkciju "UnosBroja" sa tri parametra. Funkcija treba da omogući pouzdano unošenje brojeva u program, uz potpunu detekciju grešaka pri unosu (uključujući i situacije kada ima *viška unijetih znakova iza broja*, poput unosa "123xy"). Prvi i drugi parametar su tipa "string". Pri tome, prvi parametar predstavlja tekst koji se ispisuje korisniku kao obavijest da treba unijeti broj (prompt), dok je drugi parametar tekst koji se ispisuje korisniku kao upozorenje u slučaju da unos nije ispravan. Treći parametar je referenca na proizvoljni numerički tip, a predstavlja promjenljivu u koju će se smjestiti uneseni broj. Na primjer, funkcija se može pozvati na sljedeći način:

   ```
   UnosBroja("Unesi prvi broj: ", "Neispravan unos!\n", prvi_broj);
   ```

   Funkcija treba da traži unos od korisnika sve dok unos ne bude ispravan, i tek nakon toga ona završava sa radom. Napisanu funkciju demonstrirajte u testnom programu koji od korisnika traži da unese realni broj x i cijeli broj n, a zatim računa i ispisuje vrijednost stepena x^n, *ne koristeći pri tome funkciju* "pow". Dijalog između programa i korisnika treba izgledati poput sljedećeg:

   ```
   Unesite bazu: neću!
   Neispravan unos, pokusajte ponovo...
   Unesite bazu: 3.215ppp
   Neispravan unos, pokusajte ponovo...
   Unesite bazu: 3.215

   Unesite cjelobrojni eksponent: 2.5
   Neispravan unos, pokusajte ponovo...
   Unesite cjelobrojni eksponent: -2

   3.215 na -2 iznosi 0.0967471
   ```

5. Napišite generičku funkciju "Presjek" koja prima dva parametra "v1" i "v2". Ovi parametri su vektori proizvoljnog ali istog tipa elemenata. Funkcija treba da kao rezultat vrati novi vektor koji se sastoji od elemenata koji se javljaju i u vektoru "v1" i u vektoru "v2" (drugim riječima, treba formirati *presjek skupova* čiji su elementi pohranjeni u vektorima "v1" i "v2"). Pored toga, u vektoru koji je vraćen kao rezultat iz funkcije svi elementi treba da budu različiti (odnosno, elemente koji se ponavljaju ne treba prepisivati više puta). Elementi pohranjeni u rezultatu treba da budu u istom redoslijedu kao što je redoslijed njihovog prvog pojavljivanja u vektoru "v1". Na primjer, ukoliko vektor "v1" sadrži redom brojeve 3, 7, 2, 6, 3, 4, 8, 1, 6 i 5, a vektor "v2" redom brojeve 4, 9, 5, 9, 7, 0, 4 i 6, rezultirajući vektor treba redom sadržavati brojeve 7, 6, 4 i 5. Napisanu funkciju testirajte u testnom programu prvo na dva vektora realnih brojeva, a zatim na dva vektora čiji su elementi stringovi (elementi se unose sa tastature). Dijalog između programa i korisnika teba izgledati poput sljedećeg:

   ```
   Test za realne brojeve...
   Unesite broj elemenata prvog vektora: 10
   Unesite elemente prvog vektora: 3 7 2 6 3 4 8 1 6 5
   Unesite broj elemenata drugog vektora: 8
   Unesite elemente drugog vektora: 4 9 5 9 7 0 4 6
   Zajednicki elementi su: 7 6 4 5

   Test za stringove...
   Unesite broj elemenata prvog vektora: 3
   Unesite elemente prvog vektora (ENTER nakon svakog unosa):
   asd www xy
   xvbb zzz ppe
   strstr kvekk
   Unesite broj elemenata drugog vektora: 3
   Unesite elemente drugog vektora (ENTER nakon svakog unosa):
   strstr kvekk
   asd www xy
   vuuvu
   Zajednički elementi su:
   asd www xy
   strstr kvekk
   ```