

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Rezultati i uvidi u kolokvije:** Rezultati će biti poslani na objavljivanje najkasnije u četvrtak, 6.9., navečer. Gdje, kada i kako će biti objavljeni **nije** nam poznato. Rezultate možete saznati osobno na početku uvida (tamo gdje su uvidi). Uvidi su u petak, 7.9., u 12 sati.

**Upute:** Na kolokviju je dozvoljeno koristiti samo pribor za pisanje i brisanje, te službeni podsjetnik. Kalkulatori, razne neslužbene tablice, papiri i sl., nisu dozvoljeni! **Mobitele isključite i spremite!** Sva rješenja napišite isključivo na papire sa zadacima, jer jedino njih predajete. Obavezno predajte **sve** papire sa zadacima, čak i ako neke zadatke niste rješavali. Ne zaboravite se **potpisati** na svim papirima! Skice smijete raditi i na drugim papirima koje će vam dati dežurni asistent. U svim zadacima **zabranjeno je korištenje dodatnih nizova** i standardne matematičke biblioteke (zaglavlje `math.h`), osim ako je u zadatku drugačije navedeno. Dozvoljeno je pisanje pomoćnih funkcija.

**Zadatak 1.** (30 bodova) Zadan je prirodni broj  $n$  (tipa `int`) i niz prirodnih brojeva `ponavljanja[]`, tipa `int`, duljine  $n$ . Za zadani prirodni broj  $x$  (tipa `int`), treba odrediti broj rastava od  $x$  na pribrojnice koji su uzlazno poredani prirodni brojevi, uz zadane kratnosti (`ponavljanja`) pojedinog pribrojnika.

Najmanji pribrojnik u rastavu se mora ponavljati točno `ponavljanja[0]` puta. Sljedeći po veličini, ako se pojavljuje u rastavu, ponavlja se točno `ponavljanja[1]` puta, itd. Ukoliko se u rastavu pojavljuje  $n$  i više različitih pribrojnika, tada se, počevši od  $n$ -tog (u uzlaznom poretku po veličini), svi ponavljaju točno `ponavljanja[n-1]` puta. Broj različitih pribrojnika u rastavu može biti bilo koji (nije zadan), tj. može biti strogo manji od  $n$ , ali i strogo veći od  $n$ .

Na primjer, za  $x=11$ ,  $n=3$  i `ponavljanja[]={3,2,1}`, funkcija treba vratiti 2, jer je:  $11 = 1 + 1 + 1 + 2 + 2 + 4$  i  $11 = 1 + 1 + 1 + 4 + 4$ . Za isti  $n$  i niz `ponavljanja[]`, te  $x=14$ , funkcija treba vratiti 4, jer je

$$\begin{aligned} 14 &= 1 + 1 + 1 + 2 + 2 + 3 + 4 \\ &= 1 + 1 + 1 + 2 + 2 + 7 \\ &= 1 + 1 + 1 + 3 + 3 + 5 \\ &= 2 + 2 + 2 + 4 + 4. \end{aligned}$$

Zadatak riješite rekurzivnom funkcijom čije argumente sami određujete, a koja vraća traženi broj rastava.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

### Zadatak 2. (20 + 5 bodova)

Cjelobrojna matrica dimenzija  $m \times n$  sprema se u memoriji kao niz nizova, tj. kao niz od  $m$  pokazivača na redove, a svaki red je niz od  $n$  elemenata tipa `int`.

- a) Napišite funkciju `int** generiraj(int m, int n)` koja prima dva prirodna broja  $m$  i  $n$ . Funkcija prvo dinamički alokira cjelobrojnu matricu dimenzija  $m \times n$ . Matricu je potrebno ispuniti vrijednostima  $f(0)$ ,  $f(1)$ ,  $\dots$ , gdje je funkcija  $f$  zadana na sljedeći način:

$$f(i) = i^3 - i^2.$$

U ovom zadatku, dijagonalom matrice smatramo neki skup ćelija koji sadrži sve ćelije s istim zbrojem indeksa (npr. skup ćelija s indeksima  $(0, 2)$ ,  $(1, 1)$ ,  $(2, 0)$  čini jednu dijagonalu). Matrica se popunjava na način da se slijeva nadesno popunjavaju sve dijagonale matrice, tj. prvo popunjavamo dijagonale čije ćelije imaju manji zbroj indeksa. Redoslijed popunjavanja unutar dijagonale je uzlazan za prvu dijagonalu, silazan za drugu, uzlazan za treću, itd.

Primjerice, matricu dimenzija  $3 \times 4$  bismo popunili na sljedeći način:

$$\begin{bmatrix} f(0) & f(1) & f(5) & f(6) \\ f(2) & f(4) & f(7) & f(10) \\ f(3) & f(8) & f(9) & f(11) \end{bmatrix}.$$

Funkcija vraća tako dobivenu matricu.

- b) Napišite funkciju `void spremi(int **a, int m, int n)` koja prima cjelobrojnu matricu `a` dimenzija  $m \times n$ . Funkcija sprema matricu `a` u novu tekstualnu datoteku `izlaz.txt`, na način da je svaki **stupac** matrice pohranjen kao **redak** izlazne datoteke.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Zadatak 3.** (25 bodova) Organizatori biciklističke utrke “Brda Andore” čuvaju podatke o osiguranim točkama utrke u vezanoj listi. Za svaku takvu točku pamti se njezino ime (string od najviše 30 znakova), udaljenost od početka utrke (double, u kilometrima), te broj dežurnih osoba na toj točki (int). Vezana lista točaka je uvijek *uzlazno* sortirana po udaljenosti od početka utrke. Možete pretpostaviti da uvijek postoji točka udaljenosti nula (start utrke) i da ne postoje dvije točke na istoj udaljenosti od starta.

- (a) Napišite definiciju odgovarajućeg tipa podatka za čvor (element) liste točaka, tako da je moguće definirati varijable naredbom `točka t`. Definirajte samo podatke koji su nužni za čuvanje takve liste u memoriji.
- (b) Napišite funkciju `ucitaj`, koja sa standardnog ulaza učitava podatke o osiguranim točkama (po jedan podatak u redu, tj. 3 reda po stvarnoj točki), sve dok ne učitava prazno ime točke = oznaka za kraj niza. Funkcija treba kreirati uzlazno sortiranu listu točaka i vratiti pokazivač na njezin početak. Točke na ulazu ne moraju biti sortirane po udaljenosti od starta utrke. Dozvoljeno je korištenje jednog polja za učitavanje stringa i funkcije `strlen`.
- (c) Zbog nepovoljne vremenske prognoze (obilna kiša), organizatori su odlučili **okrenuti** smjer utrke, tako da utrka starta iz prvobitno planiranog cilja, a završava u prvobitno planiranom startu. Napišite funkciju `okreni`, koja okreće naopako zadanu vezanu listu točaka prvobitne utrke. Funkcija treba vratiti pokazivač na početak okrenute liste. Udaljenosti od novog starta moraju biti korektne.

Napomena: Algoritam mora imati linearnu složenost u ovisnosti o duljini polazne liste.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Zadatak 4.** (15 + 15 bodova) Ulazni string  $s$ , duljine  $\leq 225$  znakova (samo slova i znamenke), se mapira u string duljine točno 225 znakova na sljedeći način:

- Znakovi ulaznog stringa se kopiraju u dvodimenzionalno polje znakova, dimenzija  $15 \times 15$ , tako da se, redom, znakovi stringa kopiraju na pozicije polja. Ukoliko u polju ostane slobodnih pozicija, na slobodne pozicije se postavljaju kopije ulaznog stringa, dok cijelo polje nije puno.
- Vanjski kvadrat dobivenog polja se rotira  $k$  puta u smjeru **kazaljke na satu**.
- Zamjenjuje se redak indeksa 0 s retkom indeksa  $j$  u polju.
- Mala slova se zamjenjuju velikim i obrnuto, te se svaka znamenka  $zn$  zamjenjuje znamenkom  $(zn + 1)\%10$ .
- Znakovi iz polja se, redom, kopiraju u mapirani string.

Npr., za ulazni string "prvi",  $k = 1$ ,  $j = 3$  i polje znakova dimenzija  $5 \times 5$ , ispravno mapirani string se dobije kao:

$$\text{"prvi"} \rightarrow \begin{bmatrix} p & r & v & i & p \\ r & v & i & p & r \\ v & i & p & r & v \\ i & p & r & v & i \\ p & r & v & i & p \end{bmatrix} \rightarrow \begin{bmatrix} r & p & r & v & i \\ v & v & i & p & p \\ i & i & p & r & r \\ p & p & r & v & v \\ r & v & i & p & i \end{bmatrix} \rightarrow \begin{bmatrix} P & P & R & V & V \\ V & V & I & P & P \\ I & I & P & R & R \\ R & P & R & V & I \\ R & V & I & P & I \end{bmatrix} \rightarrow \text{"PPRVVVVIPPIIPRRRPRVIRVIPI"}$$

- Napišite funkciju `mapiraj`, koja implementira zadani proces mapiranja stringa. Pretpostavite da je ulazno polje alocirano izvan funkcije.
- Napišite funkciju `provjeri` koja, za dani ulazni string  $ulaz$ , i binarnu datoteku otvorenu za čitanje  $f$ , provjerava je li string  $ulaz$  već mapiran zadanim postupkom. Datoteka  $f$  sadrži mapirane stringove, a nakon svakog stringa je zapisana veličina ulaznog (originalnog) stringa (jedan integer). Ukoliko je originalni string  $ulaz$  već prije mapiran (zapisan u datoteku  $f$ ), funkcija treba vratiti vrijednost 1. Inače treba zapisati mapiranu verziju stringa  $ulaz$  i odgovarajuću duljinu originalnog ulaza na kraj binarne datoteke  $f$  i vratiti vrijednost 2. Smijete alocirati memoriju za jedno dvodimenzionalno polje dimenzija  $15 \times 15$ .

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Rezultati i uvidi u kolokvije:** Rezultati će biti poslani na objavljivanje najkasnije u četvrtak, 6.9., navečer. Gdje, kada i kako će biti objavljeni **nije** nam poznato. Rezultate možete saznati osobno na početku uvida (tamo gdje su uvidi). Uvidi su u petak, 7.9., u 12 sati.

**Upute:** Na kolokviju je dozvoljeno koristiti samo pribor za pisanje i brisanje, te službeni podsjetnik. Kalkulatori, razne neslužbene tablice, papiri i sl., nisu dozvoljeni! **Mobitele isključite i spremite!** Sva rješenja napišite isključivo na papire sa zadacima, jer jedino njih predajete. Obavezno predajte **sve** papire sa zadacima, čak i ako neke zadatke niste rješavali. Ne zaboravite se **potpisati** na svim papirima! Skice smijete raditi i na drugim papirima koje će vam dati dežurni asistent. U svim zadacima **zabranjeno je korištenje dodatnih nizova** i standardne matematičke biblioteke (zaglavlje `math.h`), osim ako je u zadatku drugačije navedeno. Dozvoljeno je pisanje pomoćnih funkcija.

**Zadatak 1.** (30 bodova) Zadan je prirodni broj `d` (tipa `int`) i niz prirodnih brojeva `kratnost[]`, tipa `int`, duljine `d`. Za zadani prirodni broj `n` (tipa `int`), treba odrediti broj rastava od `n` na pribrojnike koji su silazno poredani prirodni brojevi, uz zadane kratnosti (ponavljanja) pojedinog pribrojnika.

Najveći pribrojnik u rastavu se mora ponavljati točno `kratnost[0]` puta. Sljedeći po veličini, ako se pojavljuje u rastavu, ponavlja se točno `kratnost[1]` puta, itd. Ukoliko se u rastavu pojavljuje `d` i više različitih pribrojnika, tada se, počevši od `d`-tog (u silaznom poretku po veličini), svi ponavljaju točno `kratnost[d-1]`-puta. Broj različitih pribrojnika u rastavu može biti bilo koji (nije zadan), tj. može biti strogo manji od `d`, ali i strogo veći od `d`.

Na primjer, za `n=14`, `d=3` i `kratnost[]={3,2,1}`, funkcija treba vratiti 2, jer:  $14 = 4+4+4+1+1$  i  $14 = 3+3+3+2+2+1$ . Za isti `d` i niz `kratnost[]`, te `n=21`, funkcija treba vratiti 3, jer je

$$\begin{aligned} 21 &= 7 + 7 + 7 \\ &= 5 + 5 + 5 + 3 + 3 \\ &= 4 + 4 + 4 + 3 + 3 + 2 + 1. \end{aligned}$$

Zadatak riješite rekurzivnom funkcijom čije argumente sami određujete, a koja vraća traženi broj rastava.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

### Zadatak 2. (20 + 5 bodova)

Matrica dimenzija  $m \times n$  sprema se u memoriji kao niz nizova, tj. kao niz od  $m$  pokazivača na redove, a svaki red je niz od  $n$  elemenata tipa `double`.

- a) Napišite funkciju `double** ispuna(int m, int n)` koja prima dva prirodna broja  $m$  i  $n$ . Funkcija prvo dinamički alokira matricu dimenzija  $m \times n$ . Matricu je potrebno ispuniti vrijednostima  $a_0, a_1, \dots$ , gdje je niz  $a$  zadan na sljedeći način:

$$a_i = i - \sqrt{i}.$$

U ovom zadatku, dijagonalom matrice smatramo neki skup ćelija koji sadrži sve ćelije s istim zbrojem indeksa (npr. skup ćelija s indeksima  $(3, 0), (2, 1), (1, 2), (0, 3)$  čini jednu dijagonalu). Matrica se popunjava na način da se slijeva nadesno popunjavaju sve dijagonale matrice, tj. prvo popunjavamo dijagonale čije ćelije imaju manji zbroj indeksa. Redoslijed popunjavanja unutar dijagonale je silazan za prvu dijagonalu, uzlazan za drugu, silazan za treću, itd.

Primjerice, matricu dimenzija  $2 \times 3$  bismo popunili na sljedeći način:

$$\begin{bmatrix} a_0 & a_2 & a_3 \\ a_1 & a_4 & a_5 \end{bmatrix}.$$

Funkcija vraća tako dobivenu matricu. Smijete koristiti funkcije iz `math.h`.

- b) Napišite funkciju `void ucitaj(double **a, int m, int n)` koja prima alociranu matricu `a` dimenzija  $m \times n$ . Funkcija u matricu `a` učitava matricu iz tekstualne datoteke `ulaz.txt`, koja je uređena tako da je svaki **stupac** matrice pohranjen kao **redak** ulazne datoteke.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Zadatak 3.** (25 bodova) Organizatori automobilističke utrke “Jezera Finske” čuvaju podatke o opasnim zavojima utrke u vezanoj listi. Za svaki takav zavoj pamti se njegovo ime (string od najviše 20 znakova), udaljenost od početka utrke (**double**, u kilometrima), te broj dežurnih osoba na tom zavoju (**int**). Vezana lista zavoja je uvijek *silazno* sortirana po udaljenosti od početka utrke. Možete pretpostaviti da uvijek postoji zavoj udaljenosti nula (start utrke) i da ne postoje dva zavoja na istoj udaljenosti od starta.

- (a) Napišite definiciju odgovarajućeg tipa podatka za čvor (**element**) liste zavoja, tako da je moguće definirati varijable naredbom **zavoj z**. Definirajte samo podatke koji su nužni za čuvanje takve liste u memoriji.
- (b) Napišite funkciju **pročitaj**, koja sa standardnog ulaza učitava podatke o opasnim zavojima (po jedan podatak u redu, tj. 3 reda po stvarnom zavoju), sve dok ne učitava prazno ime zavoja = oznaka za kraj niza. Funkcija treba kreirati silazno sortiranu listu zavoja i vratiti pokazivač na njezin početak. Zavoji na ulazu ne moraju biti sortirani po udaljenosti od starta utrke. Dozvoljeno je korištenje jednog polja za učitavanje stringa i funkcije **strlen**.
- (c) Zbog nepovoljne vremenske prognoze (obilan snijeg), organizatori su odlučili **okrenuti** smjer utrke, tako da utrka starta iz prvobitno planiranog cilja, a završava u prvobitno planiranom startu. Napišite funkciju **obrni**, koja okreće naopako zadanu vezanu listu zavoja prvobitne utrke. Funkcija treba vratiti pokazivač na početak okrenute liste. Udaljenosti od novog starta moraju biti korektne.

Napomena: Algoritam mora imati linearnu složenost u ovisnosti o duljini polazne liste.

## Programiranje 2 – popravni kolokvij, 4. 9. 2018.

**Zadatak 4.** (15 + 15 bodova) Ulazni string  $s$ , duljine  $\leq 225$  znakova (samo slova i znamenke), se mapira u string duljine točno 225 znakova na sljedeći način:

- Znakovi ulaznog stringa se kopiraju u dvodimenzionalno polje znakova, dimenzija  $15 \times 15$ , tako da se, redom, znakovi stringa kopiraju na pozicije polja. Ukoliko u polju ostane slobodnih pozicija, na slobodne pozicije se postavljaju kopije ulaznog stringa, dok cijelo polje nije puno.
- Vanjski kvadrat dobivenog polja se rotira  $k$  puta u smjeru **obratnom od kazaljke na satu**.
- Zamjenjuje se stupac indeksa 0 sa stupcem indeksa  $j$  u polju.
- Mala slova se zamjenjuju velikim i obrnuto, te se svaka znamenka  $zn$  zamjenjuje znamenkom  $(zn + 1)\%10$ .
- Znakovi iz polja se, redom, kopiraju u mapirani string.

Npr., za ulazni string "prvi",  $k = 1$ ,  $j = 3$  i polje znakova dimenzija  $5 \times 5$ , ispravno mapirani string se dobije kao:

$$\begin{array}{c}
 \text{"prvi"} \rightarrow \begin{bmatrix} p & r & v & i & p \\ r & v & i & p & r \\ v & i & p & r & v \\ i & p & r & v & i \\ p & r & v & i & p \end{bmatrix} \rightarrow \begin{bmatrix} r & v & i & p & r \\ p & v & i & p & v \\ r & i & p & r & i \\ v & p & r & v & p \\ i & p & r & v & i \end{bmatrix} \rightarrow \begin{bmatrix} P & V & I & R & R \\ P & V & I & P & V \\ R & I & P & R & I \\ V & P & R & V & P \\ V & P & R & I & I \end{bmatrix} \rightarrow \text{"PVIRRPVIPVRIPRIVPRVPVPRII"}
 \end{array}$$

- (a) Napišite funkciju `mapiraj`, koja implementira zadani proces mapiranja stringa. Pretpostavite da je ulazno polje alocirano izvan funkcije.
- (b) Napišite funkciju `provjeri` koja, za dani ulazni string  $ulaz$ , i binarnu datoteku otvorenu za čitanje  $f$ , provjerava je li string  $ulaz$  već mapiran zadanim postupkom. Datoteka  $f$  sadrži mapirane stringove, a nakon svakog stringa je zapisana veličina ulaznog (originalnog) stringa (jedan integer). Ukoliko je originalni string  $ulaz$  već prije mapiran (zapisan u datoteku  $f$ ), funkcija treba vratiti vrijednost 1. Inače treba zapisati mapiranu verziju stringa  $ulaz$  i odgovarajuću duljinu originalnog ulaza na kraj binarne datoteke  $f$  i vratiti vrijednost 2. Smijete alocirati memoriju za jedno dvodimenzionalno polje dimenzija  $15 \times 15$ .