

**CS103 - DOMAĆI ZADATAK 06.**

Prilikom slanja domaćeg zadatka svom asistentu neophodno je da ispunite sledeće:

* Subject mail-a mora biti CS103-DZbr. Za ovaj domaći zadatak - CS103-DZ06
* Sve fajlove, koji su deo rešenja zadataka, arhivirati (zip, rar, …)
* Poželjno je uraditi i printscreen koda pre pokretanja programa
* U prilogu maila treba da se nalazi arhiva projekta koji se ocenjuje, imenovana na sledeći način: CS103-DZbr-BrojIndeksa-ImePrezime*. Na primer, CS103-DZ06-9999-VeljkoGrkovic.zip*
* Telo mail-a treba da ima pozdravnu poruku

**Molimo sve studente da se pridržavaju navedenog, inače zadaci neće biti pregledani i ocenjeni.**

Studenti iz Beograda i online studenti zadatke na ocenjivanje šalju mail-om na adresu [lazar.mrkela@metropolitan.ac.rs](mailto:lazar.mrkela@metropolitan.ac.rs)

Studenti iz Niša zadatke na ocenjivanje šalju mail-om na adresu  
[nikola.dimitrijevic@metropolitan.ac.rs](mailto:nikola.dimitrijevic@metropolitan.ac.rs)

**Rok za slanje domaćih zadataka:**

* **Tradicionalni studenti: sedam dana od dana održavanja vežbe na koju se odnosi domaći zadatak. Nakon toga, broj poena se umanjuje za 50%. Krajnji rok za slanje rešenja je deset dana pre termina ispita u kome želite da polažete ispit.**
* **Studenti na online nastavi, kao i studenti kojima je predmet prenet u narednu godinu ili je diferencijalni: najkasnije deset dana pre termina ispita u kome želite da polažete ispit.**

*Svaki student radi jedan zadatka sa spiska*. Ostali zadaci mogu da posluže za vežbanje i pripremu ispita, ali ih ne šaljete na pregled asistentima.

Redni broj zadatak dobijate tako što vaš broj indeksa podelite sa 19, a dobijeni ostatak pri deljenju uvećate za 1. Na primer:   
Broj indeksa 2378  
2378 % 19 + 1 = 4 – student radi 4. zadatak.

1. Napisati program koji učitava niz karaktera u jednoj liniji (dok se ne pređe u novi red), a korišćenjem steka ispisuje karaktere u inverznom poretku. Koristiti jednostruko povezanu listu za implementaciju steka.
2. Napisati program koji učitava niz celih brojeva u rastućem poretku i korišćenjem steka kao jednostruko povezane liste ispisuje brojeve u opadajućem poretku. Unos brojeva prekinuti kada se naruši rastući poredak tj. kada se desi da je sledeći uneti broj manji od prethodnog.
3. Neka je ***Q*** neprazan red, a ***S*** prazan stek. Napisati program i algoritam koji korišćenjem steka ***S*** vrši inverziju članova reda ***Q.***
4. Napisati program i metodu za implementaciju reda pomoću niza, korišćenjem dve pomoćne promenljive ***front*** i ***rear***, takve da ukoliko ***rear*** dostigne kraj niza, ceo red se pomera ka početku, tj. dok se front ne pomeri do indeksa 0.
5. Napisati program i metod za implementaciju ***deka*** korišćenjem kružnog niza. Generisati klase ***Deque*** i ***Queue*** takve da je klasa Deque izvedena iz klase Queue.
6. Napisati program za rad sa stekom. Stek implementirati pomoću jednostruko povezane liste. Napisati metodu ***size(…)*** za stek takvu da se brojanje vrši pomeranjem pomoćnog pokazivača dok se ne stigne do dna steka.
7. Napisati program za rad sa stekovima i redovima korišćenjem povezanih listi (definisati klasu za stek i red i odgovarajuće metode). Napisati program i metode koje:
   1. pomeraju sve članove steka u red.
   2. pomeraju sve članove reda u stek.
8. Napisati program za rad sa stekovima korišćenjem povezanih listi (definisati klasu za stek i odgovarajuće metode). Napisati program I metode koje:
   1. prebacuju podatke iz jednog steka u drugi stek tako da podaci ostanu u istom redosledu.
   2. prebacuju podatke iz jednog steka u drugi stek tako da podaci ostanu u inverznom redosledu.
9. Napisati program za rad sa stekovima i redovima korišćenjem povezanih listi (definisati klasu za stek I red I odgovarajuće metode). Napisati program I metode koje:
   1. korišćenjem lokalnog steka invertuju redosled članova u redu.
   2. korišćenjem lokalnog reda invertuju redosled članova u steku.
10. Napisati funkciju koja će iz steka izbaciti sve one brojeve koji se pojavljuju dva ili više puta. Funkcija ima sledeći prototip:   
     ***void izbaci\_duple(Stek \*stek);*** ili ***public void izbaci\_duple(Stek stek);***
11. Napisati funkciju koja će iz steka izbaciti sva pojavljivanja broja n. Funkcija mora u novi stek, koji je inicijalno prazan, ubaciti brojeve koji predstavljaju koliko je broj bio udaljen od originalnog vrha steka. Funkcija ima sledeći prototip:   
     ***void izbaci\_broj(Stek \*stek, int n, Stek \*novi);* ili *public void izbaci\_broj(Stek stek, int n, Stek novi);***
12. Napisati funkciju koja će na osnovu ulaznih stekova stek1 i stek2 stvoriti novi stek koristeći pritom sledeće pravilo da se za novi element u novom steku uvek odabere manji od elemenata na vrhu stekova stek1 i stek2. Ukoliko se jedan od stekova isprazni, onda uzeti preostale elemente iz drugog. Ulazni stekovi moraju nakon završetka funkcije ostati nepromenjeni. Funkcija ima prototip   
     ***Stek \*spoji(Stek \*stek1, Stek \*stek2); ili  
     public Stek spoji(Stek stek1, Stek stek2);***
13. Zadate su funkcije za stavljanje i skidanje elemenata sa stek i za stavljanje i skidanje elemenata iz reda (realizovanih listom):   
    ***int dodajNaStek (int element, atom \*\*glava);   
    int skiniSaSteka (int \*element, atom \*\*glava);  
    int dodajURed (int element, atom \*\*glava, atom \*\*rep);  
    int skiniIzReda (int \*element, atom \*\*glava, atom \*\*rep);***   
    ili

***public int dodajNaStek (int element, Stek glava);   
public int skiniSaSteka (int element, Stek glava);  
public int dodajURed (int element, Stek glava, Stek rep);  
public int skiniIzReda (int element, Stek glava, Stek rep);***  
  
Implementirati gore navedene metode.   
Napišite rekurzivnu funkciju, koristeći gore navedene funkcije, koja prebacuje elemente sa steka u red, prototipa:   
***void prebaci (atom \*\*glavaSteka, atom \*\*glavaReda, atom \*\*repReda);*** *ili*

***public void prebaci (Stek glavaSteka, Red glavaReda, Red repReda)***  
Početni stek mora ostati očuvan. Redosled elemenata u redu isti je kao na steku (onaj element koji prvi izlazi sa steka je prvi element koji izlazi i iz reda). Red je na početku prazan.

1. U kontekstu strukture/generičke klase stek napišite funkciju   
    ***void insert\_second (elementtype x, stek \*s);  
    public void insert\_second (<T> x, Stek s);*** koji ubacuje element x u stek s, tako da x bude drugi element odozdo (tj. drugi element od dna steka). Međusobni redosled ostalih elemenata ostaje nepromenjen.
2. Napišite ne-rekurzivnu funkciju  
    ***int SearchQ(queue \*q); ili  
    public int SearchQ(Queue q);***   
   koji pretražuje red od početka do kraja i pritom izbacuje duplikate elemenata koji su se već pre pojavili u redu. Uz to, broji koliko puta se koji element pojavio u redu i vraća najveći takav broj. Međusobni redosled ostalih (ne-izbačenih) elemenata ostaje nepromenjen.
3. Date su funkcije za stavljanje i skidanje elemenata iz reda realizovanog listom:   
   ***DodajURed (element, glava, rep);   
   SkiniIzReda (\*element, glava, rep);***  
   Napišite funkciju, koristeći gore navedene funkcije, koja stvara (i preko argumenata funkcije vraća) novi red koji sadrži samo negativne elemente iz zadanog reda. Funkcija ima prototip:   
   ***RedNegativnih(glavaZadani, repZadani, glavaTrazeni, repTrazeni);***Početni red mora ostati očuvan. Možete koristiti pomoćni red.
4. Posmatrajmo implementaciju “dinamičkog” steka pomoću niza promenljive dužine. Stek sadrži celobrojne podatke. Potrebno je napisati funkcije za stavljanje elementa na stek i skidanje elementa sa steka. Ako na steku nema mesta za novi element, stek treba dinamički povećati za 10 novih mesta i upisati novi element. Takođe, kod skidanja elementa sa steka, potrebno je proveriti koliko ima praznih mesta. Ako ima više od 10 praznih mesta, veličinu steka potrebno je smanjiti za 10. Funkcije trebaju vratiti 1 ako je operacija uspela, a 0 inače. Napišite i deklaraciju tipa podatka ***Stack***.
5. U kontekstu strukture stack napišite funkciju   
    ***void insert\_N (elementtype x, Stack \*s, int n);  
    public void insert\_N (<T> x, Stack s, int n);***koji ubacuje element X u stek S, tako da X bude n-ti element odozdo (tj. n-ti element od dna steka). Međusobni redosled ostalih elemenata ostaje nepromenjen.
6. Dat je stek S u kojem se čuvaju podaci tipa char. Na žalost, neki od elemenata steka su specijalni karakteri (npr. \*). Napišite ne-rekurzivnu funkciju sa prototipom   
    ***void ukoni\_spec (Stack s);*** ili  
    ***public void ukoni\_spec (Stack s);***  
   koja će iz steka S ukloniti sve zvezdice, a ostale elemente poređati u ”naopakom” poretku. Na primer, ako su elementi u S redom od vrha prema dnu steka (\*, D, \*, C, B, A), onda nakon poziva funkcije stek treba izgledati ovako: (A, B, C, D).