Konstrukcija i analiza algoritama 2 (prvi kolokvijum, smer R)

1. a) Konstruisati AVL stablo od brojeva 100, 132, 134, 170, 180, 112, 188, 184, 181, 165 **(2 poena)**
b) Konkatenacija je operacija nad dva skupa koji zadovoljavaju uslov da su svi ključevi u jednom skupu manji od svih ključeva u drugom skupu; rezultat konkatenacije je unija skupova. Konstruisati algoritam za konkatenaciju dva AVL stabla u jedno. Složenost algoritma u najgorem slučaju treba da bude O(h), gde je h veća od visina dva AVL stabla. **(2 poena)**
c) Odrediti najmanji broj čvorova koje može imati AVL stablo visine h. **(2 poena)**

2. Zadat je težinski bipartitni graf sa *k* čvorova i *l* grana. Kritična težina uparivanja u grafu G je težina najteže grane u uparivanju. Konstruisati algoritam složenosti $O(\sqrt{k}(k+l)\log l)$ za nalaženje optimalnog uparivanja sa minimalnom kritičnom težinom. **(6 poena)**

3. Konstruisati (u pseudo-kodu ili C/C++) algoritam vremenske složenosti O(n) koji će sortirati niz celih brojeva iz skupa {-5, 3, 4}. Dimenzija niza je n. **(6 poena)**

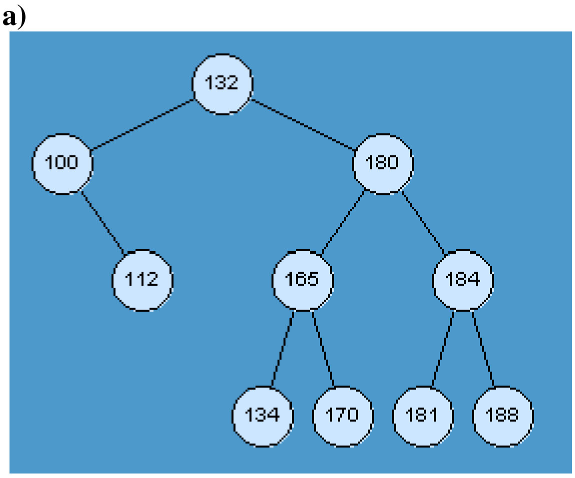
4. Da li su sledeća tvrđenja tačna? Obrazložiti netačna tvrđenja primerom ili tačnim tvrđenjem.
a) U Graham algoritmu za konstrukciju konveksnog omotača, tačke se smeštaju u red po FIFO principu.
b) Day–Stout–Warren algoritam ima linearnu memorijsku i vremensku složenost.
c) U neusmerenom grafu sa 2n čvorova nije moguće naći optimalno uparivanje ako je stepen svakog čvora ne manji od n.
d) Svako maksimalno uparivanje je i optimalno uparivanje.
e) Hamiltonov ciklus grafa G sadrže sve grane grafa G. **(5*1+1 poen)**

5. Matricom povezanosti zadat je težinski graf G sa 6 čvorova. Odrediti maksimalan protok u ovom usmerenom grafu od polaznog čvora 0 do ishodišta (čvora 5). **(6 poena)**

$$\begin{bmatrix} 0 & 10 & 18 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 0 & 0 \\ 0 & 0 & 0 & 16 & 6 & 4 \\ 0 & 0 & 0 & 0 & 2 & 10 \\ 0 & 0 & 0 & 0 & 0 & 18 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

RESENJA

1.
a)



b) Neka je B_h oznaka za AVL stablo visine h sa najmanjim brojem čvorova sa i neka je $\mathbf{N_h}$ oznaka za broj čvorova stabla B_h .

B_h ima BAR JEDNO podstablo visine h-1 (stablo B_{h-1}), a zbog svojstva da B_h je AVL stablo sa najmanjim brojem čvorova, njegovo drugo podstablo je B_{h-2}, te važi:

$N_h=N_{h-1}+N_{h-2}+1$
$N_{h+1}=N_h+N_{h-1}+1$
$N_{h+1}- N_h = N_h +N_{h-1}+1- N_{h-1}- N_{h-2}-1$
$N_{h+1} - 2N_h + N_{h-2}=0$
Karakteristična jednačina ove rekurentne veze je
$t^3-2t^2+1=0$
odnosno $(t-1)(t^2-t-1)=0$
odnosno njeni koreni su $t_1=1$, $t_2=\frac{1+\sqrt{5}}{2}$, $t_3=\frac{1-\sqrt{5}}{2}$
Zato je opsti član niza N_h jednak
$N_h = C_1 + C_2 * (\frac{1+\sqrt{5}}{2})^h + C_3 * (\frac{1-\sqrt{5}}{2})^h$

Zbog $N_0=1$, $N_1=2$, $N_2=4$
dobija se sistem:
$1=C_1 + C_2 + C_3$
$2= C_1 + C_2 * \frac{1+\sqrt{5}}{2}+ C_3 * \frac{1-\sqrt{5}}{2}$
$4= C_1 + C_2 * (\frac{1+\sqrt{5}}{2})^2 + C_3 * ((\frac{1-\sqrt{5}}{2})^2$
Rešenja sistema su:
$C_1 = - 1$
$C_2 = \frac{2+\sqrt{5}}{\sqrt{5}}$
$C_3 = \frac{\sqrt{5}-2}{\sqrt{5}}$

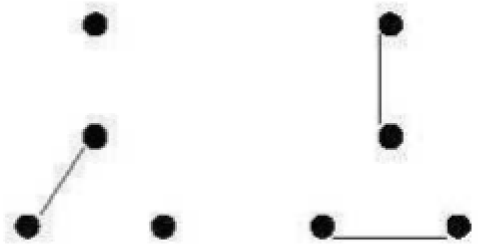
c) Neka su zadata dva AVL stabla T, G, tako da svi ključevi stabla G su veći od svih ključeva stabla T.
Uz čvorove stabla T i G čuvaće se i informacija o visini, jer je to svojstvo značajano za AVL stabla, kao tip balansiranih stabala i neka je, na primer, visina h_2 stabla G sada manja ili jednaka od visine h_1 stabla T. Drugi slučaj rešava se analogno.
1. Ukloniti najveći čvor iz stabla T, na primer čvor r i on će biti koren stabla preko kog se stablo G privezuje za T. Evo kako.
2. Spuštajući se u stablu T samo desnim granama, naći čvor v visine h_2 ili h_2-1 (a to je moguće jer h1>=h2). Neka otac čvora v jeste čvor p.
3. Postaviti da desni sin čvora p ne bude cvor v, vec čvor r iz koraka 1. (što je održava poredak, jer čvor r je sa najvećim ključem, te može biti desni sin čvora p)
4. Postaviti da levi sin čvora r bude čvor v sa svojim podstablom iz T, a desni sin je koren stabla G.
To je u redu sa stanovista BSP svojstva.
OBRAZLOŽENJE: r je najveci čvor u T, te ključ čvora v nije veći od njega i može čvor v biti levi sin,a kako svaki čvor stabla G, te i koren je veći od najvećeg ključa iz T, onda koren iz G može biti desni sin sa ciljem da i novoformirano stablo bude binarno stablo pretrage (BSP).
5. U koracima 3 i 4 formirano je stablo koje ima BSP svojstva, ali možda nije uravnoteženo, tj. visina čvora r može da postane veća za 1 od visine čvora v koji je bio na tom mestu, te je nuzno obaviti eventualno uravnotezenje pomocu rotacije.
Sve operacije traju <=O(h1) koraka zbog svojstva AVL stabla.

2. Koristićemo kombinaciju binarne pretrage i algoritma za nalaženje optimalnog uparivanja.
Rešavamo najpre malo drugačiji problem: pitamo se da li za dato *x* postoji optimalno uparivanje takvo da su težine svih njegovih grana manje ili jednake od *x*
Ovaj problem možemo rešiti tako što iz grafa uklonimo sve grane čije su težine veće od *x* i zatim proveravamo da li optimalno uparivanje u novodobijenom smanjnom grafu ima jednak broj grana kao optimalno uparivanje u polaznom grafu.
Složenost ove provere jednaka je složenosti algoritma za nalaženje optimalnog uparivanja u grafu koje je O($\sqrt{k}(l+k)$).
U grafu G ima *l* grana, te ima najviše *l* različitih težina. Binarnom pretragom tražimo najmanje *x* tako da je *x* težina neke grane i da postoji optimalno uparivanje u kome sve grane imaju težinu manju ili jednaku *x*. Zato je ukupna složenost O($\sqrt{k}(k+l)\log l$).

3.

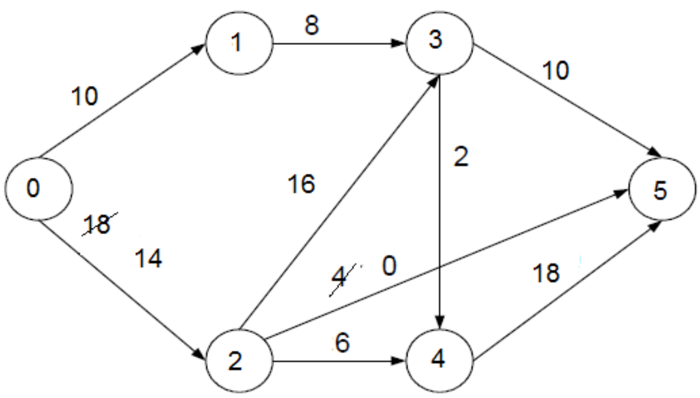
```
/* program count2;*/
#include<stdio.h>
#include<limits.h>
main()
{
int i,j,n,min,max,k;    int a[200],o[900];
scanf("%d",&n);
min=INT _MAX;max=0; k=0;
for (j=0;j<=900;j++) o[j]=0;
for(i=1;i<=n;i++)
 {   scanf("%d",&a[i]); o[a[i]]+=1;
  if (a[i]>max) max=a[i];
  if (a[i]<min) min=a[i];
 }
 for (i=min;i<=max;i++) if (o[i]!=0) for (j=1;j<=o[i];j++)
      {
         k=k+1;  a[k]=i;
       }
for (i=1;i<=n;i++) printf("%d ",a[i]);
}
```

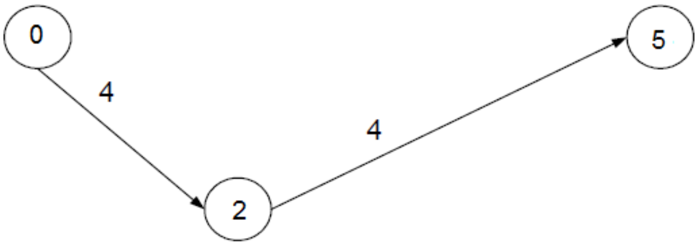
4.
a) Ne. U Graham algoritmu za konstrukciju konveksnog omotača, tačke se smeštaju u stek po LIFO principu.
b) Ne. Day–Stout–Warren algoritam ima linearnu vremensku složenost i O(1) memorijsku slozenost.
c) Ne. U neusmerenom grafu sa 2n čvorova moguće naći optimalno uparivanje ako je stepen svakog čvora bar n. (videti algoritam 6.9.1 u udzbeniku)
d) Ne. Svako maksimalno uparivanje ne mora biti i optimalno uparivanje. Evo primera



e) Ne. Hamiltonov ciklus grafa G sadrže sve čvorove grafa G.

5.
1. Uzmimo, na primer, put koji čine čvorovi 0–2–5 (voditi računa o smeru kojim se može proći iz jednog čvora u drugi). Grana koja spaja čvorove 0 – 2 ima težinu 18, grana koja spaja 2 – 5 ima težinu 4. Dakle, najviše što može proteći ovim putem jeste 4 (zbog grane koja spaja 2 – 5).
Da bi lakše vodili evidenciju o tome šta smo poslali ka čvoru 5 i koliko još možemo poslati nekom granom koja je učestvovala u putu kojim se već nešto šalje, crtamo dva grafa (vidi sliku). Jedan u kojem precrtavamo puteve kojima šaljemo i ažuriramo podatak o tome koliko smo poslali (slika donja), a drugi u kojem ćemo vršiti ažuriranje protoka grana koje su učestvovale u nekom putu (slika gornja).

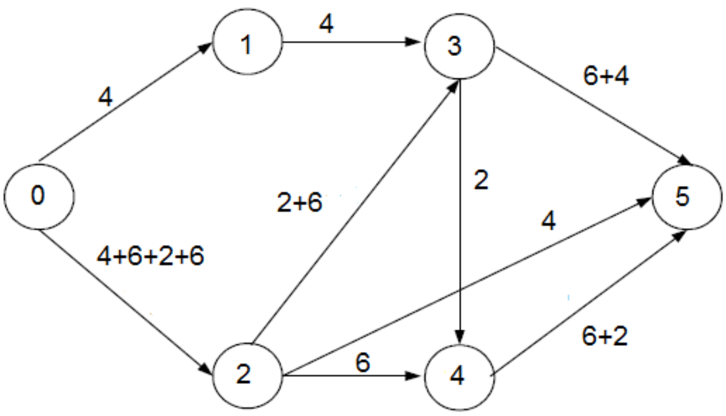




Na osnovu slike 2 gore, zaključujemo da će nova težina za granu koja spaja čvorove 0–2 biti 14 i da kao takva može biti deo nekog novog puta. Dalje, težina grane koja spaja čvorove 2–5 će biti 0. Ova grana više ne može učestvovati ni u jednom putu. Jedan od sledećih puteva bi mogao biti 0 – 2 – 4 – 5. Od 0 – 2 možemo poslati 14, od 2 – 4 može proteći 6, a od 4 – 5, protok je 18. Zaključujemo da datim putem možemo poslati najviše 6. Skiciramo novu sliku.

Zaključujemo da ni granu 2 – 4 ne možemo više koristiti. Kroz granu 4 – 5 može proteći još 6, kroz granu koja ide od 0 – 2 možemo poslati 8, pa gledamo da li postoji još neki put koji sadrži ove dve grane. Postoji put, 0–2–3–4–5.
Protoci su im 8 – 16 – 2– 12. Šaljemo najviše što možemo datim putem, dakle 2, ažuriramo protoke za grane koje čine put 0 –2 – 3 – 4 – 5 na 6 – 14 – 0 – 10.
Zaključujemo da se granom 3 – 4 više ništa ne može poslati.
Od čvora 0 se može poslati preko grane 0–2 još 6, pa gledamo da li postoji neki put kojim se može stići od 0 do 5. Put je 0 – 2 – 3 – 5. Težine grana ovog puta su 6 – 14 – 10. Šaljemo 6, ažurirane vrednosti za grane su 0 – 8 – 4. Skiciramo novu sliku.

Vidimo da od čvora 0 ne možemo više ići do čvora 5 preko čvora 2. Gledamo da li postoji neki put od čvora 0 do čvora 5 preko čvora 1 sa protokom različitim od 0. Očigledno je da je jedini takav put 0 – 1 – 3 – 5.
Odgovarajući protoci su 10 – 8 – 4, dakle šaljemo 4, novi protoci su 6 – 4 – 0.
Ažuriramo podatke i dobijamo sliku.



Sa slike vidimo da se iz čvora 0 može poslati još 4 preko čvora 1 u čvor 3. Međutim, ne postoji grana ili put kojim se čvor 3 može povezati sa čvorom 5, a da ima nenulti protok. Dakle, došli smo do kraja.
Ukupan protok dobijamo sabiranjem onoga što utiče u čvor 5 u konačnoj slici gore. Maksimalan protok će biti: 6 + 4 + 4 + 6 + 2 = 22.
Dobijeni putevi su samo jedan od mogućih načina da se obezbedi maksimalan protok u najvećem broju slučajeva. Rešenje u pogledu puteva kojima se taj protok ostvaruje nije jedinstveno. Maksimalan dobijeni protok mora biti isti bez obzira na to koje smo puteve odabrali ukoliko je algoritam izvršen do kraja.

Jedan od načina da se odredi vizuelno da smo došli do kraja algoritma jeste da se odstrane sve grane čiji je protok jednak nuli i nacrta dobijeni graf (vidi sliku dole). Ako nakon uklanjanja grana sa nultim protokom ne postoji put od početnog do krajnjeg čvora, algoritam je završen.