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

1. Konstruisati AVL stablo i regularnu skip listu koji redom sadrži brojeve sa ulaza od 1 do 17. **(2+2 poena)**

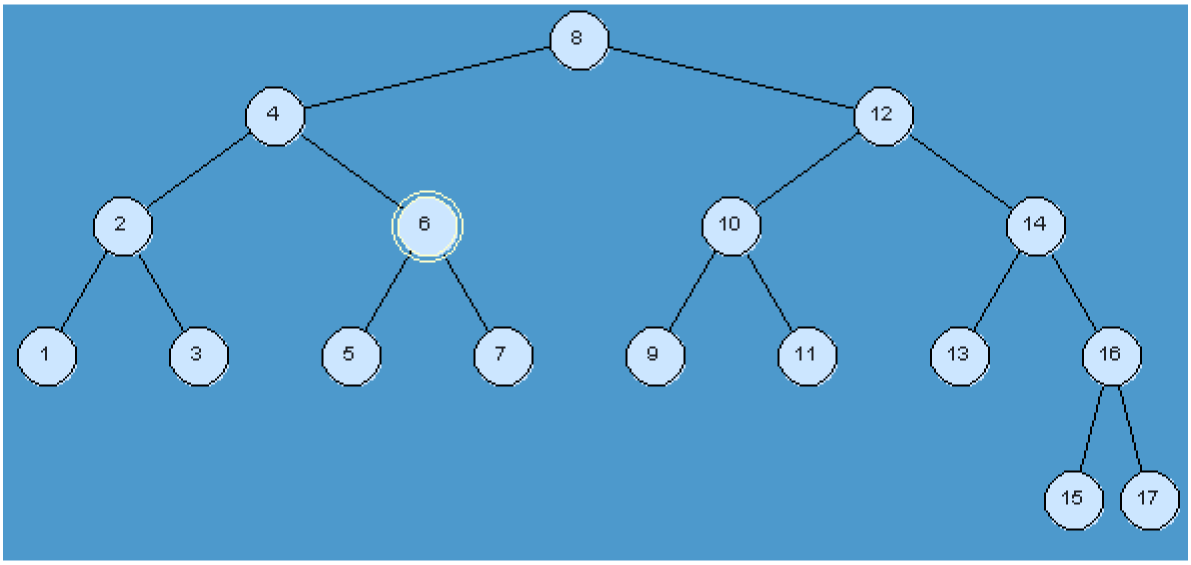
2. Konstruisati (u pseudo-kodu ili C/C++) algoritam vremenske složenosti O(n) koji učitava sa standardnog ulaza niz realnih brojeva a, dimenzije n i pronalazi i ispisuje na standardni izlaz član niza čiji broj pojava u nizu je veći od n/4.
Na primer u nizu: 4 5 6 7 7 6 5 4 7 6 5 4 5 rešenje je 5 **(4 poena)**

3. Konstruisati (u pseudo-kodu ili C/C++) algoritam vremenske složenosti O(n) koji će sortirati niz ocena studenata (5<=ocena<=10). Dimenzija niza je n>=10000. **(4 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) Vremenska složenost pretraživanja u AVL stablu sa n čvorova je O(n log n) pošto su AVL stabla uvek visinski balansirana
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 poena)**

5. Dokazati: Graf G je bipartitan akko ne sadrži neparne cikluse. **(3 poena)**

REŠENJA
1.



2.
```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100
#define NUM 4
int main()
{
   int a[MAX],i,n,j,k,prazno,x[NUM],p[NUM];
   printf("Unesite dimenziju niza! ");
   scanf("%d",&n);
   printf("\nUnesite clanove niza! ");
   for(i=0;i<n;i++) scanf("%d",a+i);
   for(j=0;j<NUM;j++) p[j]=-1;
   for(i=0;i<n;i++)
   {
     prazno=-1;
     for(j=0;j<NUM;j++)
     {
        if ((p[j]>=0) && (x[j]==a[i])) p[j]+=3;
        else p[j]-=1;
        if(p[j]<=0) prazno=j;
     }
     if(prazno>=0)
     {x[prazno]=a[i];
       p[prazno]=3;
     }
```

```
  }
  for(j=0;j<NUM;j++) p[j]=0;
 for(i=0;i<n;i++)
   for(j=0;j<NUM;j++)
     if(a[i]==x[j]) p[j]++;
   k=(n/4)+1;
   for(j=0;j<NUM;j++)
     if(p[j]>=k)
     {printf("%d ",x[j]); return 0;}
  printf("\nNe postoji takav element\n");
   return 0;
}
```

3.

```
#include<stdio.h>
#include<limits.h>
#define MAX 10000
main()
{
int i,j,n,min,max,k;
int a[MAX],o[MAX];
scanf("%d",&n);
min=10;max=5; k=0;
for (j=0;j<MAX;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. Vremenska složenost pretraživanja u AVL stablu sa n čvorova je O(log n) pošto su AVL stabla uvek visinski balansirana
c) Ne. U neusmerenom grafu sa 2n čvorova moguće je naći savršeno uparivanje ako je stepen svakog čvora bar n. (poglavlje 6.9 udžbenika)
d) Ne. Važi obrnuto: svako optimalno uparivanje je maksimalno uparivanje.
e) Ne. Hamiltonov ciklus grafa G sadrže sve čvorove grafa G.

5.
Da! Dokaz:

(=>) Trivijalno. Neka je graf G bipartitan sa biparticijom (A, B). Podjimo od nekog cvora a iz A. Svaki put neparne duzine dovodi nas do cvora u B, a svaki put parne do cvora u A. Kako treba da se vratimo u a da bi napravili ciklus, on mora biti parne duzine.
Drukčije napisano:
Neka je G bipartitan graf s biparticijom (X, Y), a $C = v_0 v_1 \ldots v_h v_0$ proizvoljan ciklus u G.
-uzmimo da je $v_0 \in X$
-zbog $v_0 v_1 \in E$ i $v_0 \in X$ sledi da je $v_1 \in Y$
-opet je $v_2 \in X$ itd..., tj. uošte važi $v_{2l} \in X$ i $v_{2l+1} \in Y$.
-Zbog $v_0 \in X$ je $v_h \in Y$, a pošto je $v_h v_0$ grana grafa, to je $h = 2l + 1$ za neko l
$\Rightarrow$ C je paran ciklus

(<=) Nesto tezi smer. Nađimo neko drvo razapinjanja u grafu G, i označimo ga sa T. Uzmemo neki, bilo koji cvor, v. Do svakog cvora mozemo iz v stici na jedinstven nacin iduci ivicama koje se nalaze u T. Sada tvrdimo da su one do kojih je put parne duzine u jednoj, a do kojih je neparne u drugoj particiji. Za svaku ivicu e (x, y) vazi jedno od sledeca 2:

(i) e se nalazi u T. U ovom slucaju je ili x pretposlednju cvor na v-y putu kroz T, ili je y pretposlednji cvor na v-x putu kroz T. U svakom slucaju duzine puteva do x i y su razlicite, pa su oni u razlicitim particijama.

(ii) e se ne nalazi u T. U ovom slucaju, x-y put u T, zajedno sa ivicom e pravi ciklus. Po *(i)* imamo da su cvorovi na x-y putu u T naizmenicno u jednoj i drugoj particiji. Posto je svaki ciklus paran, onda su x i y u razlicitim particijama.