Fakultet tehničkih nauka, Novi Sad
Animacija u inženjerstvu

17. II 2016. godine

Diskretne i kombinatorne metode za računarsku grafiku

Dat je algoritam

```
1: function PARTITION(A, p, r)
2:     x ← A[r]
3:     i ← p − 1
4:     for j ← p to r − 1 do
5:         if A[j] ≤ x then
6:             i ← i + 1
7:             exchange(A[i], A[j])
8:         end if
9:     end for
10:    exchange(A[i + 1], A[r])
11:    return i + 1
12: end function
```

1. Posle primene algoritma PARTITION$(A, 1, 7)$ na ulaz $A = [5, 1, 8, 2, 9, 6, 3]$, koje će biti stanje niza A?

2. Koliko poređenja (linija 5) će biti izvršeno na ulaznom nizu iz zadatka 1?

   Koliko puta će se pozvati procedura exchange (linije 7 i 10) za ulazni niz iz zadatka 1 i koliko puta će se zamena u exchange izvršiti?

3. Napisati rekurzivnu proceduru SORT(A, p, r) koja bi korišćenjem procedure PARTITION komandom SORT$(A, 1, 7)$ uradila Quick sort sortiranje niza A.

   procedure SORT(A, p, r)

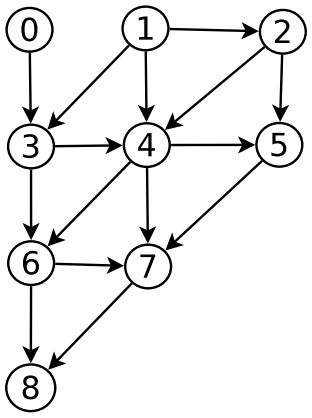
   end procedure

4. Dati definiciju "velikog O" ponašanja i pokazati da je $n \ln n + n = O(n^2)$.

   Da li je $n \ln n - n = O(n)$?

   Da li je $n\sqrt{n} + n = O(n^2)$?

   Da li je $\frac{3}{4}n^2 - 3n \ln n = O(n^2)$?



5. Napisati kod funkcije enqueue_list koja unosi čvor na kraj liste susedstva grafa. Napisati deo koda za unos grafa sa slike u okviru procedure main u niz listi susedstva G leksikografski.

6. Na graf sa slike primeniti DFS algoritam, kod čvorova napisati d i f vrednosti, na grane napisati tip, nacrtati šumu DFS-a. Ako je dati graf usmereni aciklični graf (DAG), dati topološko sortiranje čvorova datog grafa.

7. U tabeli su date cene prevoza između 5 gradova.

   (a) Polazeći od čvora 1, metodom najjeftinijeg suseda naći približno rešenje problema trg. putnika (TSP).

   (b) Za isti problem naći Mađarskom metodom angažovanje koje je rešenje relaksiranog TSP.

   (c) Znajući rešenja (a) i (b), naći granice optimalnog rešenja.

	1	2	3	4	5
1	-	8	12	13	4
2	8	-	4	9	9
3	14	5	-	10	6
4	12	8	12	-	7
5	5	9	7	6	-

```
#define max_cv 50
typedef struct _node gnode;
typedef gnode *grana;
struct _node
{
    int data;
    gnode *next;
};

void enqueue_list(grana **gp, cvor d)
{
// Ovaj kod napisati
}
```

```
int main(void)
{
    grana G[max_cv],GT[max_cv];
    int i,n; grana *rear[max_cv];

    for(i=0;i<max_cv;i++){
        G[i] = NULL;
        rear[i] = &(G[i]);
    }
    enqueue_list(&rear[0],3);
// nastaviti dalje,
// uneti ostatak grafa
    return 0;
}
```

Bodovi: 1→15, 2→5, 3→10, 4→10, 5→15, 6→20, 7→15.