Fakultet tehničkih nauka, Novi Sad
Animacija u inženjerstvu

28. X 2018. godine

Diskr. i komb. met. za rač. gr.

1→15, 2→5, 3→10, 4→10, 5→5, 6→15, 7→15, 8→15.

1. Napisati algoritam za sortiranje biranjem, takozvani SELECTION SORT.

Za algoritam SELECTION SORT iz zadatka 1, za niz dužine n, neka je $S(n)$ broj zamena i $P(n)$ broj poređenja.

2. Za niz $[8, 1, 2, 3, 4, 5, 6, 7]$ naći $S(n)$ i $P(n)$.

3. Koliko je $S(n)$ i $P(n)$ za najgori slučaj ulaznog niza dužine n algoritma SELECTION SORT?

4. Dati definiciju "malog o" ponašanja i pokazati da je $\frac{1}{4}\, n \ln n + 40\, n - 10 = o(n^2)$.

   Za niz dužine n neka je $T_{WM}(n)$ najgori slučaj vremena sortiranja Merge sort algoritmom i $T_{WQ}(n)$ najgori slučaj vremena sortiranja Quick sort algoritmom. Da li je $T_{WM}(n) = o(T_{WQ})$?

   Da li je $\frac{3}{4}\, n^2 + 3\, n \sqrt{n} = o(n^2)$?

   Da li je $\frac{3}{4} n + 3\sqrt{n} = o(n)$?

---

5. Nacrtati usmereni graf G koji je dat tabelom listi susedstva:

u	Adj(u)
0	
1	4, 5, 7
2	1, 6
3	0, 1, 4, 7
4	0, 7
5	0
6	4, 5
7	0, 5

6. Na graf G primeniti DFS algoritam, kod čvorova napisati d i f vrednosti, kod grana napisati tip (TBCF), napraviti tabelu zagrada. Ako je dati graf usmereni aciklični graf (DAG), dati topološko sortiranje čvorova.

7. Napisati kod funkcije enqueue_list koja unosi čvor na kraj liste susedstva grafa. Napisati deo koda za unos grafa G (iz zad. 5) u okviru procedure main u niz listi susedstva grafa G leksikografski.

---

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

   (a) Polazeći od čvora 4, metodom najjeftinijeg suseda naći približno rešenje problema trg. putnika (TSP).

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

	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[1],4);
    // nastaviti dalje,
    // uneti ostatak grafa
    return 0;
}
```