

Tema 2

Termen limita: joi, 24.11.2016, ora 18:00

Exercițiul 1. Reprezentare prin ADN

Un informatician a descoperit un nou mod de a codifica un număr întreg folosind acidul dezoxiribonucleic, ce conține A (adenină), C (citozină), G (guanină) și T (timină).

Astfel, în calcule se folosește baza de numerație 4, dar cifrele A, C, G și T au valorile A=0, C=1, G=2, iar T=-1.

De exemplu, numărul întreg 27 se reprezintă, folosind ADN, prin "GTT". Asemănător, numărul întreg -6 se reprezintă prin "TGG", iar numărul 84 se reprezintă prin "CCCA".

Justificare:

$$GTT = G \times 4^2 + T \times 4^1 + T \times 4^0 = 2 \times 16 + (-1) \times 4 + (-1) \times 1 = 32 - 4 - 1 = 27$$

$$TGG = T \times 4^2 + G \times 4^1 + G \times 4^0 = (-1) \times 16 + 2 \times 4 + 2 \times 1 = -16 + 8 + 2 = -6$$

$$CCCA = C \times 4^3 + C \times 4^2 + C \times 4^1 + A \times 4^0 = 1 \times 64 + 1 \times 16 + 1 \times 4 + 0 \times 1 = 84$$

Se consideră declarația

```
#define MAX_ADN 35
```

Cerințele exercițiului:

a. (2p) Scrieți o funcție

```
void codificareADN(long long nrIntreg, char sirADN[MAX_ADN])
```

care codifica un număr întreg nrIntreg în reprezentarea ADN.

De exemplu, pentru nrIntreg=27, funcția va returna sirADN="GTT".

b. (1p) Scrieți o funcție

```
long long decodificareADN(char sirADN[MAX_ADN])
```

care decodifica un șir ADN într-un număr întreg, de tip long.

De exemplu, pentru sirADN="GTT" funcția va returna 27.

c. (1p) Scrieți o funcție

```
void adunareADN(char primulSir[MAX_ADN], char alDoileaSir[MAX_ADN], char suma[MAX_ADN])
```

care adună două șiruri ADN, rezultatul fiind reprezentarea în ADN a sumei celor două numere, reprezentate de primulSir și alDoileaSir.

De exemplu, pentru primulSir="GTT" și alDoileaSir="C", funcția va obține suma="GTA".

d. (1p) Scrieți o funcție

```
void scadereADN(char primulSir[MAX_ADN], char alDoileaSir[MAX_ADN], char diferenta[MAX_ADN])
```

care scade două șiruri ADN, rezultatul fiind reprezentarea în ADN a sumei celor două numere, reprezentate de primulSir și alDoileaSir.

§ De exemplu, pentru primul Sir="GTA" și alDoileaSir="C", funcția va obține diferența="GTT".

Observatii: timpul maxim de execuție este 0,5 secunde pentru fiecare funcție și fiecare test. Niciun cod nu va începe cu „A” (cu excepția codului numărului 0).

Exercițiul 2. O fată și un băiat

O fată și un băiat joacă următorul joc de echipă.

Fata dispune de o serie de cartonașe dreptunghiulare, în care, în partea de sus apare o literă mare a alfabetului englezesc (‘A’...’Z’), iar în partea de jos apare una sau două litere ale alfabetului englezesc (‘A’..’Z’).

Exemple de cartonașe pe care le poate avea fata:

B DM	C Z	C BD	A AM	F NN
---------	--------	---------	---------	---------

Băiatul dispune de o serie de cartonașe dreptunghiulare, în care, în partea de sus apare o literă mare a alfabetului englezesc (‘A’...’Z’), iar în partea de jos apare o cifră (‘0’..’9’).

Exemple de cartonașe pe care le poate avea băiatul:

B 9	C 0	C 1	A 3	F 3
--------	--------	--------	--------	--------

Separat, într-o cutie, există o mulțime suficient de mare de cartonașe pătrate, mici, pe care sunt scrise atât cifrele de la ‘0’ la ‘9’, cât și literele de la ‘A’ la ‘Z’ și care se pot repeta de mai multe ori. Pentru derularea jocului, există un număr suficient de cartonașe cu fiecare cifră sau literă.

La începutul jocului, cei doi iau din cutie mai multe cartonașe cu cifre, formând un șir de cifre, numit **T**. (**T** are cel puțin o cifră).

Exemple de șiruri **T**:

1 8 3 0 1 2	0 0 8 9 1 1	2 8 3 3 5 9 1 6	3 2 9 7 4 4 1 0
-------------	-------------	-----------------	-----------------

Jocul constă în obținerea, de către fată și băiat (împreună) a acestui șir de cifre **T**, folosind cartonașele date și plecând de la un șir **S**, care inițial este format dintr-o singură literă (de exemplu, **S** este “A” sau **S** este “T”).

Cei doi jucători fac mutări, dar NU neapărat alternativ, ci de comun acord, cu scopul de a obține **T**, plecând de la **S**, folosind următoarele proceduri.

Fata poate înlocui în **S** orice literă pe care o are pe un cartonaș în zona galbenă cu cele două litere pe care le are în zona portocalie. Ea poate face acest lucru de 0, 1 sau mai multe ori, re folosind cartonașele,

Băiatul poate înlocui în **S** orice literă pe care o are pe un cartonaș în zona galbenă cu cifra pe care o are în zona verde. El poate face acest lucru de 0, 1 sau mai multe ori, re folosind cartonașele.

Scopul lor este de a obține pe **T**, plecând de la **S**, folosind orice mutare de înlocuire posibilă, de oricâte ori este nevoie.

Exemplu de joc:

Fata dispune de:

A	B	B	C	C
BC	D	ED	F	FB

Băiatul dispune de:

E	D	D	F	E
0	2	3	4	1

Arbitrul stabilește șirul de oprire **T** ca fiind

2	4	0	3
---	---	---	---

pornire **S** ca fiind “A”, deci:

A

Iată cum pot juca cei doi jucători, pentru a obține pe **T**, plecând de la **S**:

1. Mai întâi fata folosește cartonașul (A,BC) și înlocuiește pe A cu BC, obținând:

B	C
---	---

2. Apoi, tot ea folosește cartonașul (B,D) și obține:

D	C
---	---

3. Acum intervine băiatul și folosește cartonașul său (D,2), obținându-se:

2	C
---	---

4. Fata vine acum cu folosirea cartonașului (C,FB) și obține:

2	F	B
---	---	---

5. Acum băiatul folosește cartonașul (F,4) și se obține:

2	4	B
---	---	---

6. Fata folosește cartonașul (B,ED) și obține șirul:

2	4	E	D
---	---	---	---

7. Acum băiatul face două mutări la rând, întâi folosind cartonașul (E,0), obținând:

2	4	0	D
---	---	---	---

8. iar apoi cartonașul (D,3), obținând șirul **T** final:

2	4	0	3
---	---	---	---

Cerin a exercițiului:

Se consideră următoarele declarații:

```
#define MAX_CARTONASE 1000
#define MAX_SIR 15

typedef struct {
    char galben;
    union {
        char verde;
        char portocaliu[3];
    };
} cartonase;

struct mutare {
    cartonase nod;
    struct mutare *urmator;
};

typedef struct mutare* lista;
```

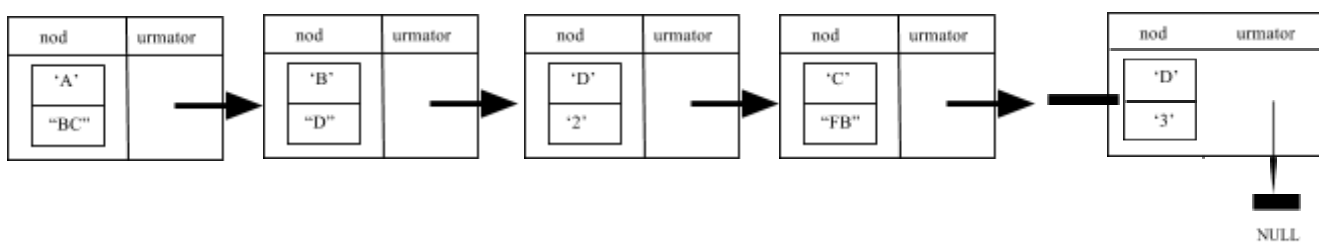
(5p) Să se scrie o funcție

```
lista mutariJoc(cartonase vecCartonase[MAX_CARTONASE], char literaStart,
char sirFinal[MAX_SIR])
```

care să returneze:

- o listă simplu înlănțuită, de lungime minimă, în care nodurile conțin mutările echipei (cartonașele folosite), dacă și numai dacă cei doi jucători pot obține șirul `sirFinal` plecând din șirul format inițial doar din caracterul `literaStart` și folosind cartonașele din vectorul `vecCartonase`,
- respectiv `NULL`, dacă cei doi jucători nu-l pot obține pe `sirFinal`.

Pentru exemplul ilustrat mai sus, funcția ar returna lista de lungime minimă:



Observatii:

Nu vor exista bucle (cartonase de genul (A,A) sau (A,B) si (B,A) sau (A,B), (B,C) si (C,A)).

Timpul maxim de executie pentru fiecare test si fiecare functie: 1 secunda.

Aveti posibilitatea sa folositi biblioteca `string.h`.