Probleme consultații – 10 noiembrie 2018

Problema 1
Joc: Număr redus

Enunț

Fie n si k două numere naturale date, n>9 (n poate avea până la 100 de cifre) și k>0 (k<numarul de cifre ale lui n). Să se elimine k cifre din numărul n astfel încât numărul rămas să fie maxim.

Exemple:

n= 9875623 si k= 3 ➔ 9876
n= 5903418 si k =4 ➔ 948
n= 5903418 si k =6 ➔ 9

Analiză

- Numarul n se memoreaza sub forma de sir de cifre 0..9 cu m cifre.
- Se va aplica de k ori
  - subalgoritmul care va cauta prima cifra mai mica decat urmatoarea
  - subalgoritmul care va elimina cifra gasita
- Exemple:
  - Pentru n = 9875623 si k=3
    - Se elimina pe rand cate o cifra
      - Elimina cifra 5: 987623
      - Elimina cifra 2: 98763
      - Elimina cifra 3: 9876
  - Pentru n = 5903418si k=4
    - Se elimina pe rand cate o cifra
      - Elimina cifra 5: 903418
      - Elimina cifra 0: 93418
      - Elimina cifra 3: 9418
      - Elimina cifra 1: 948

Specificarea funcțiilor

Funcția **detIndexProprietate (m,a,i)**:
Descriere: Returneaza indexul din vectorul a cu proprietatea a[i]<a[i+1].
Date: m – lungime vector a, a –vectorul de cifre, i-pozitia de verificat.
Rezultate: pozitia din vectorul a cu proprietatea ca a[i]<a[i+1]

Varianta recursivă

Modelul matematic:

$$\boldsymbol{detIndexProprietate(m,a,i)}\begin{cases} \boldsymbol{m, daca\ i = m} \\ \boldsymbol{i, daca\ a[i] < a[i+1]} \\ \boldsymbol{detIndexProprietate(m,a,i+1), altfel} \end{cases}$$

Metoda **stergeElementPozitieData (m,a,indexP)**:
Descriere: Sterge din vectorul a elemental de pe pozitia indexP.
Date: m – lungime vector a, a –vectorul (de cifre), indexP-pozitia elementului ce trebuie sters.
Rezultate: m-lungime vector a, a – vectorul modificat

Metoda **reduLaNumarMaximDekOri (m,a,k)**:
Descriere: Reduce numarul dat la un numar maxim prin eliminarea a k cifre.
Date: m – lungime vector a, a –vectorul (de cifre), k-numarul de cifre ce trebuie eliminate.
Rezultate: m-lungime vector a, a – vectorul modificat

Implementare

Varianta Pascal

```
Program Joc_NumarRedus;
type vector = array [ 1..50] of integer;

procedure citireSir(var m:integer; var a:vector);
var i:integer;
begin
  readln(m);
  for i:=1 to mdo
  begin
    read(a[i]);
  end;
end;
procedure afisareSir(m:integer; a:vector);
var i:integer;
begin
  for i:=1 to m do
  begin
    write(a[i]);
  end;
end;

function detIndexProprietate(m:integer;a:vector;i:integer):integer;
begin                                    {recursiv, se autoapeleaza functia}
    if(i<=(m-1)) then
        if (a[i]>=a[i+1]) then
            detIndexProprietate:=detIndexProprietate(m,a,i+1)
        else
            detIndexProprietate:=i
    else
        if (i=m) then
            detIndexProprietate:=m;
end;
```

```
procedure stergeElementPozitieData(var m:integer; var a:vector; indexP:integer);
var p:integer;    //sterge elementul de pe pozitia indexP din vector
begin
       for p:=indexP to m-1 do
           a[p]:=a[p+1];
       m:=m-1;
end;

procedure reduLaNumarMaximDekOri(var m:integer;var a:vector; k:integer);
var j,indexP:integer;
begin
for j:=1 to k do        // se elimina de k ori cate o cifra cu proprietatea cautata
  begin
    if(m>=1) then     //daca numarul are cel putin 1 cifra
    begin
        indexP:=detIndexProprietate(m,a,1);
        if (indexP=m) then
                    m:=m-1
             else
                    stergeElementPozitieData(m,a,indexP);
       end;
  end;
end;

var m,k:integer;
    a:vector;
begin
   citireSir(m,a);
   readln(k);
   write('numarul citit='); afisareSir(m,a);
   writeln;
   reduLaNumarMaximDekOri(m,a,k);
   write('numar redus=');
   afisareSir(m,a);
end.
```

Varianta C++

```
#include <iostream>

using namespace std;

void citireSir(int &m, int a[]){
  cin>>m;
  for (int i=0;i<m;i++){
      cin>>a[i];
  }
}

void afisareSir(int m, int a[]){
  for (int i=0;i<m;i++){
    cout<<a[i];
  }
}
```

```
int detIndexProprietate(int m,int a[], int i){
    if(i<=(m-1))
        if (a[i]>=a[i+1])
            return detIndexProprietate(m,a,i+1);
        else
            return i;
    else
        if (i==m)
            return m;
}

void stergeElementPozitieData(int  &m,int a[], int indexP){
       int p;  //sterge elementul de pe pozitia indexP din vector
       for (p=indexP;p<=m-1;p++)
           a[p]=a[p+1];
       m=m-1;
}

void reduLaNumarMaximDekOri(int &m,int a[],int k){
  int j,indexP;
  for (j=1;j<=k;j++)        // se elimina de k ori cate o cifra cu proprietatea cautata
    if(m>=1){      //daca numarul are cel putin 1 cifra
        indexP=detIndexProprietate(m,a,0);
        if (indexP==m)
                    m=m-1;
             else
                    stergeElementPozitieData(m,a,indexP);
      }
}


int main(){
int m=0,k=0;
int a[100];

   citireSir(m,a);
   cin>>k;
   cout<<"numarul citit="; afisareSir(m,a);
   cout<<endl;
   reduLaNumarMaximDekOri(m,a,k);
   cout<<"numar redus=";
   afisareSir(m,a);
    return 0;
}
```

Exemple

Date de intrare	Rezultate
m=7,a=[5,9,0,3,4,1,8], k=4	948
m=7,a=[5,9,0,3,4,1,8], k=6	9
m=7,a=[9,8,7,5,6,2,3], k=3	9876
m=7,a=[9,8,7,5,6,2,3], k=4	987
m=7,a=[9,8,7,5,6,2,3], k=5	98

Problema 2
Joc: Păsărică mută-ți cuibul

Enunț

Se dau n cuiburi pe care se află n-1 copii astfel încât fiecare copil se află într-un cuib. În fiecare moment un copil poate schimba cuibul în care stă în cuibul care este liber. Se dă ordinea inițială și ordinea finala a copiiilor și se cere să se determine ordinea în care vor schimba cuiburile copiii pentru a ajunge la configurația finală. Cuibul în care nu se afla nici un copil este notat cu 0.

Exemple:

n=4

Configuratia initiala: 3 2 0 1

Configuratia finala: 2 1 3 0

Solutie:

Pasul 1) copilul 3 pleaca de la cercul 1 la cercul 3: 0 2 3 1
Pasul 2) Copilul 2 pleaca de pe cercul 2 pe cercul 1: 2 0 3 1
Pasul 3) Copilul 1 pleaca de pe cercul 4 pe cercul 2: 2 1 3 0

Analiză

- Se va porni de la configuratia initiala si pentru fiecare pozitie i a caror elemente nu corespund in cele doua configuratii, se va proceda in doi pasi astfel:
  - Pasul a) se va elibera spatiu in configuratia initiala, respectiv 0 pe pozitie
  - Pasul b) se va cauta pozitia din cofiguratia initiala pe care se gaseste elementul din configuratia finala de pe pozitia i
- Modificarile de mai sus se vor aplica cat timp exista cel putin o pozitie cu elemente diferinte in cele 2 configuratii.
- Exemplu:

Configuratia initiala 2 0 3 4 5 1

Configuratia finala 5 4 1 0 3 2

Mutari:

2,0,3,4,5,1,
0,2,3,4,5,1,
5,2,3,4,0,1,
5,0,3,4,2,1,
5,4,3,0,2,1,
5,4,0,3,2,1,
5,4,1,3,2,0,
5,4,1,0,2,3,
5,4,1,0,2,3,
5,4,1,2,0,3,
5,4,1,2,3,0,
5,4,1,0,3,2,

Specificarea funcțiilor

Funcția **cautaIndice (m,ci,cf,i)**:
Descriere: Returneaza indexul din vectorul ci cu proprietatea pe care se aflat elementul cf[i]
Date: m – lungime vector,ci, cf –vectorul de configuratie initiala si finala, i-pozitia de verificat.
Rezultate: pozitia j din vectorul ci cu proprietatea ca cf[i]=ci[j]

Functia **oPerecheDiferita (m,ci,cf)**:
Descriere: Returneaza true/false daca exista cel putin o pozitie pentru care elementele din ci sic f sa fie distincte.
Date: m – lungime vector , ci, cf –vectorul de configuratie initiala si finala.
Rezultate: true- exista cel putin o pozitie cu elemente disticte, false – toate elementele ci sic f sunt la fel pe aceleasi pozitii

Metoda **MutariJoc (m,ci,cf,nm,mm)**:
Descriere: Determina mutarile de la configuratie initiala ci la configuratia finala cf.
Date: m – lungime vector a, , ci, cf –vectorul de configuratie initiala si finala.
Rezultate: nm –numarul de mutari, mm-matrice cu mutarile realizate

Implementare

Varianta Pascal

```
Program Joc_Pasarica_muta_ti_cuibul;
type vector = array [ 1..50] of integer;
type matrice = array [1..50,1..50] of integer;

var n,i,j,k,l,nm:integer;
    ci,cf:vector;
    b,q:boolean;
    mm:matrice;

procedure citireSir(var n:integer; var ci:vector);
begin
  readln(n);
  for i:=1 to n do
  begin
    read(ci[i]);
  end;
end;

procedure afisareSir(n:integer; ci:vector);
var i:integer;
begin
  for i:=1 to n do
  begin
    write(ci[i],', ');
  end;
end;
```

```
procedure afisareMatrice(n,nm:integer; mm:matrice);
var i,j:integer;
begin
  for i:=1 to nm do
  begin
    for j:=1 to n do
            write(mm[i,j],', ');
    writeln;
  end;
end;


function cautaIndice(n:integer; ci:vector; cf:vector; i:integer):integer;
{Descriere: Cauta si returneaza indicele j din ci pe care se afla elementul cf[i]}
var b:boolean;
begin
j:=1; b:=false;
writeln;
while(j<=n) and not b do
    if cf[i]=ci[j] then b:=true
    else j:=j+1;
cautaIndice:=j;
end;

function oPerecheDiferita(n:integer; ci,cf:vector):boolean;
var q:boolean;
    j:integer;
begin
    q:=true;     j:=1;
    while (j<=n) and (q) do
        begin
            if ci[j]<>cf[j] then q:=false;
            j:=j+1;
        end;
    oPerecheDiferita:=q;
end;

procedure MutariJoc(n:integer; ci:vector;cf:vector; var nm:integer; var mm:matrice);
var k,j:integer;
    q:boolean;
begin
  for i:=1 to n do   //determina pozitia libera in configuratia initiala, adica k
    if ci[i]=0 then
       k:=i;
  nm:=1;
  for i:=1 to n do
        mm[1,i]:=ci[i];

  q:=false; i:=1;
  while not q do {Cattimp exista cel putin o pereche diferita intre ci si cf}
  begin
    if(ci[i]<>cf[i]) then
        begin
            if (ci[i]<>0)  then
                begin
                    ci[k]:=ci[i];
                   ci[i]:=0;   nm:=nm+1;
                    for l:=1 to n do mm[nm,l]:=ci[l];
                end;
```

```
            {cauta pozitia j din ci pe care se afla elementul cf[i]}
            j:=cautaIndice(n,ci,cf,i);
    {muta pe pozitia libera din ci elementul "la fel" din cf, adica de pe pozitia j din ci}
            ci[i]:=ci[j];
           // writeln('Copilul ', ci[j], ' pleaca de pe cercul ', j,' pe cercul ',i);
            ci[j]:=0;
            {noua pozitie libera k este j}
            k:=j;
            nm:=nm+1;
            for l:=1 to n do mm[nm,l]:=ci[l];

        end;
    i:=i+1;
    q:=oPerecheDiferita(n,ci,cf);
  end;

end;

begin
  citireSir(n,ci);
  citireSir(n,cf);
  MutariJoc(n,ci,cf,nm,mm);
  writeln('Afisare mutari:');
  afisareMatrice(n,nm,mm);
  end.
```

Varianta C++

```
#include <iostream>

using namespace std;

void citireSir(int &m, int a[]){
//  cout<<"Dati m="<<endl;
  cin>>m;
  for (int i=1;i<=m;i++){
     // cout<<endl<<"Dati numarul a["<<i<<"]=";
      cin>>a[i];
  }
}

void afisareSir(int m, int a[]){
  for (int i=1;i<=m;i++){
    cout<<a[i];
  }
}

void afisareMatrice(int n, int nm, int mm[50][50]){
  for (int i=1;i<=nm;i++){
    for (int j=1;j<=n;j++)
            cout<<mm[i][j]<<",";
    cout<<endl;
  }
}

int cautaIndice(int n, int ci[], int cf[], int i){
//Descriere: Cauta si returneaza indicele j din ci pe care se afla elementul cf[i]
bool b;
int j=1;
b=false;
```

```
while((j<=n) and (not b))
    if (cf[i]==ci[j])
        b=true;
    else
        j=j+1;
return j;
}

bool oPerecheDiferita(int n, int ci[], int cf[]){
    bool q; int j;
    q=true;
    j=1;
    while ((j<=n) and (q)){
        if (ci[j]!=cf[j])
             q=false;
        j=j+1;
    }
    return q;
}

void MutariJoc(int n, int ci[50], int cf[50], int &nm, int mm[50][50]){
int k,j,i; bool q;
 //determina pozitia libera in configuratia initiala, adica k
  for (int i=1;i<=n;i++)
    if (ci[i]==0)
       k=i;
  nm=1;
  for (int i=1;i<=n;i++)
        mm[1][i]=ci[i];
  cout<<endl;

  q=false; i=1;
  //Cattimp exista cel putin o pereche diferita intre ci si cf
  while (not q){
    if(ci[i]!=cf[i]) {
            if (ci[i]!=0) {
                    ci[k]=ci[i];
                    ci[i]=0;
                // writeln('copilul ', ci[i],' pleaca de la cercul ', i, ' la cercul ',k);
                    nm=nm+1;
                    for (int l=1;l<=n;l++)
                        mm[nm][l]=ci[l];
            };
            //cauta pozitia j din ci pe care se afla elementul cf[i]
            j=cautaIndice(n,ci,cf,i);
    //muta pe pozitia libera din ci elementul "la fel" din cf, adica de pe pozitia j din ci
            ci[i]=ci[j];
           // writeln('Copilul ', ci[j], ' pleaca de pe cercul ', j,' pe cercul ',i);
            ci[j]=0;
            //noua pozitie libera k este j
            k=j;
            nm=nm+1;
            for (int l=1;l<=n;l++)
                mm[nm][l]=ci[l];
    };
    i=i+1;
    q=oPerecheDiferita(n,ci,cf);
  }
 }
```

```
int main(){
int n=0,nm;
int ci[50],cf[50], mm[50][50];
  citireSir(n,ci);
  citireSir(n,cf);
  MutariJoc(n,ci,cf,nm,mm);
  cout<<"Afisare mutari:"<<endl;
  afisareMatrice(n,nm,mm);
    return 0;
}
```

Exemple

Date de intrare	Rezultate
n=6, ci=[2 0 3 4 5 1],cf=[5 4 1 0 3 2]	2,0,3,4,5,1,
	0,2,3,4,5,1,
	5,2,3,4,0,1,
	5,0,3,4,2,1,
	5,4,3,0,2,1,
	5,4,0,3,2,1,
	5,4,1,3,2,0,
	5,4,1,0,2,3,
	5,4,1,2,0,3,
	5,4,1,2,3,0,
	5,4,1,0,3,2,

Problema 3
Bile rostogolite
Enunț

In planul xOy se rostogolesc din dreapta spre stânga, rând pe rând, mai multe cercuri (de raze conoscute / date) care se opresc la axa Oy sau cand ating alt cerc.

a) Care este **marginea dreaptă** limită (ocupată de cercuri)?

b) Care dintre cercuri pot fi eliminate fără ca (astfel încât) marginea dreaptă să (nu) se modifice?

Pentru exemplul din figura de mai jos, pot fi eliminate cercurile 1,2, 4.

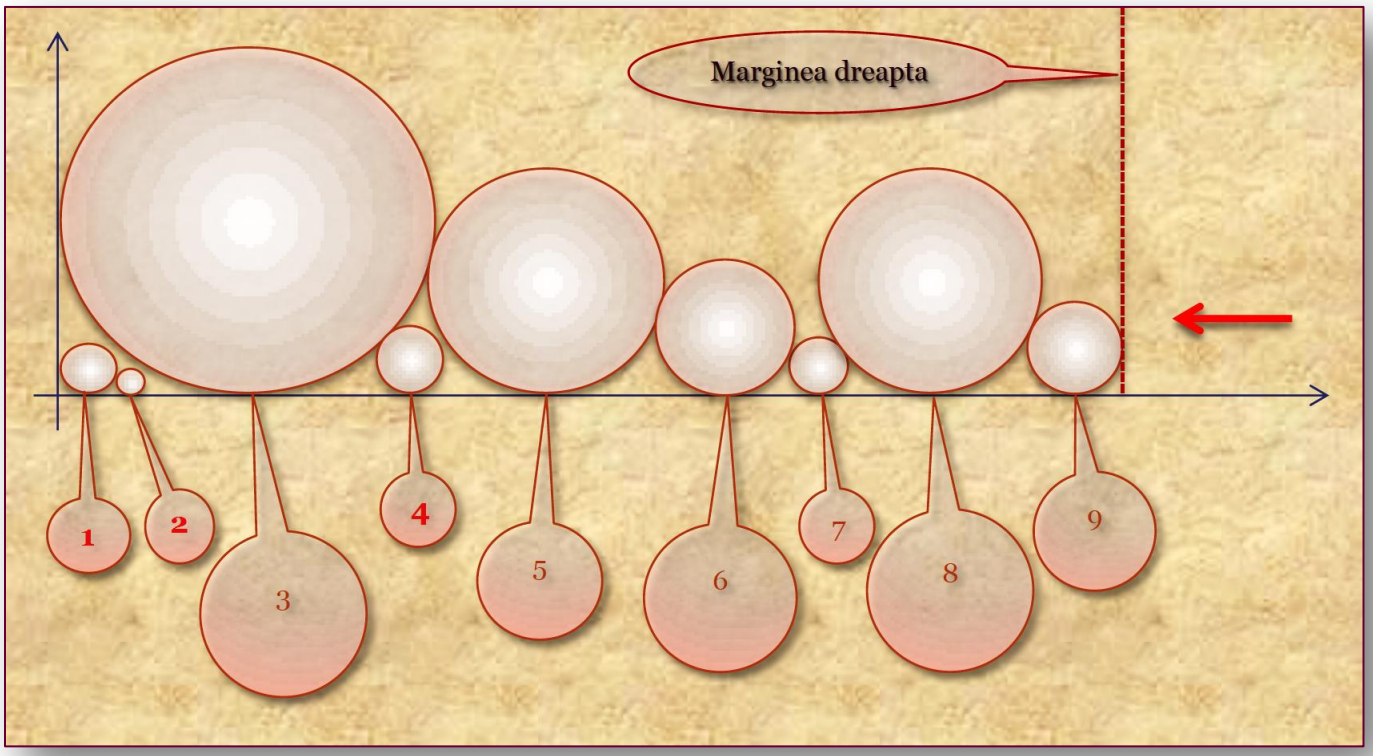

Fig. 1. Bile Rostogolite

Analiză

Pentru fiecare cerc C_i se cunoaşte raza lui (r_i) şi vom determina poziţia lui finală (x_i). Distanţa dintre două poziţii a două cercuri tangent este data în Fig. 2.

In triunghiul desenat cu roşu se poate observa că

$$d^2 = (r_1+r_2)^2 - (r_1-r_2)^2 = 4 * r_1 * r_2$$

Aceasta înseamnă ca vom calcula $x_2 = x_1+d$.

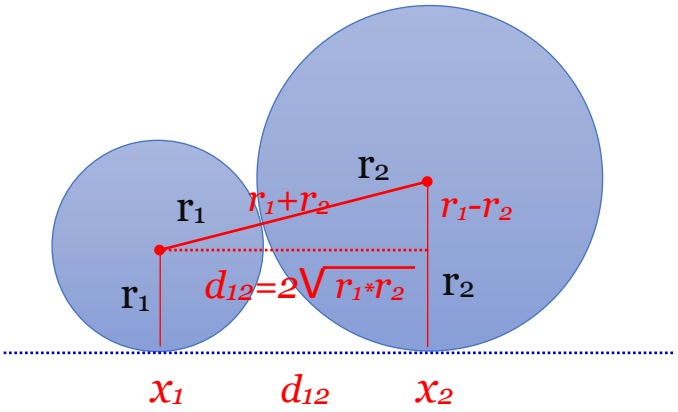

Fig. 2 Două cercuri tangente

Vom calcula valorile $x_1, x_2, \ldots, x_n$ astfel:

- $x_1 = r_1$
- $x_i = \underset{j=1,i-1}{Max}(Max(x_j + d_{ji}), r_i), \quad i = \overline{2,n}$

Observaţie : Dacă cercul tangent p nu este predecesorul lui i ($p<i-1$), atunci toate cercurile $x_{p+1}, \ldots, x_{i-1}$ se pot elimina. De asemenea pot fi eliminate şi ultimele cercuri $C_{p+1}, C_{p+2}, \ldots, C_n$ dacă $Md = x_p + r_p$ iar $p<n$.

- **Marginea dreaptă** se poate calcula după formula:

$$\boldsymbol{Md} = \underset{i=1,n}{Max}(x_i + r_i)$$

Exemplu.

Pentru razele (2, 3, 1, 7, 2, 10, 1) rezultatele sunt:

a) ***Marginea dreaptă*** este **42.7973**,

b) Cercurile care pot fi eliminate sunt **3, 5, 7**.

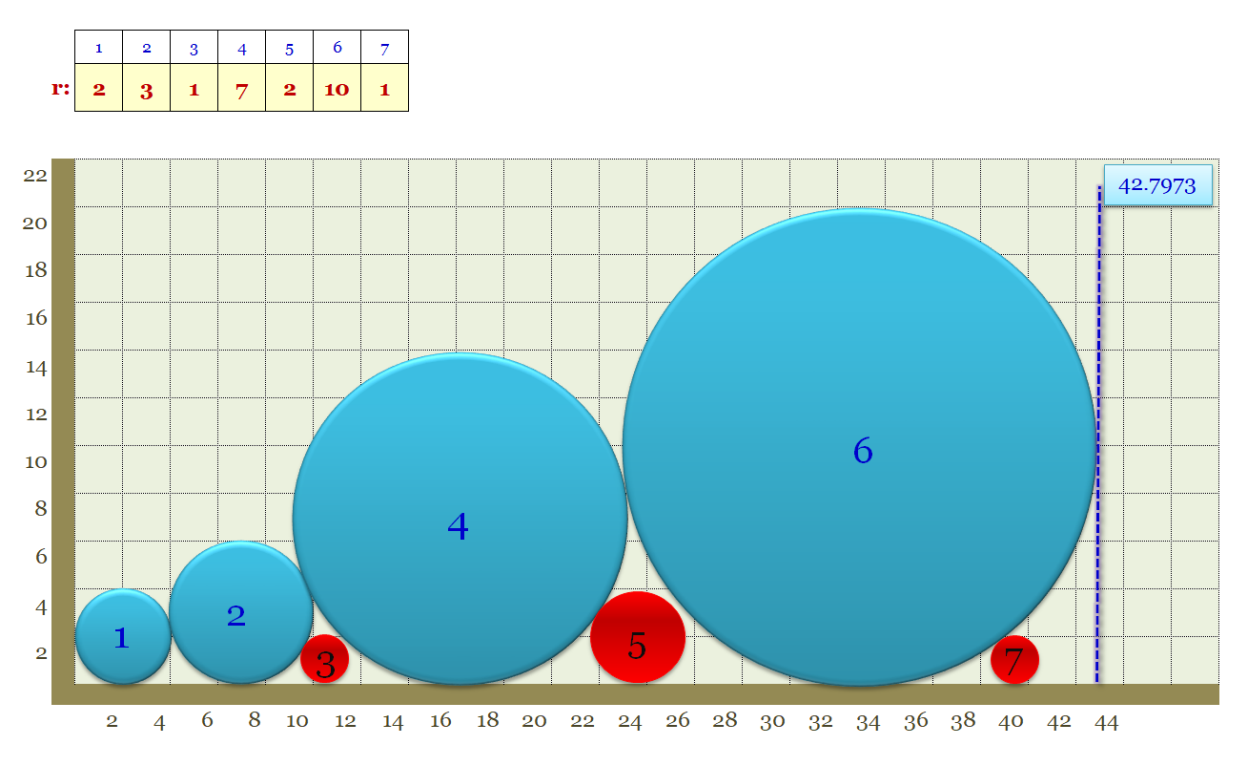

Fig. 3 Rostogolile spre stânga

Specificarea funcțiilor *(Descriere: ... , Date: ..., Rezultate: ...)*

- `d(r,j,i)` - Calculează distanța d_{ij} descrisă în Fig. 2 (d = *Distanța* (Cj,Ci) = $2\sqrt{r_i * r_j}$;
- `PozX(r,x,j,i)` - Determină pozitia *cercului i* dacă ar fi tangent la *cercul j*;
- `Rost_B(r,x,El)` - Determină pozițiile finale x_i ale cercurilor (de raze r_i), precum şi mulțimea *El* a cercurilor care pot fi eliminate;
- `Max(r,x,n)` - Returnează valoarea Max{(xi+ri)/$i=\overline{1,n}$ } reprezintând *marginea dreaptă*
- `Md(r,x)` – Determină *Marginea dreaptă*;
- `Print(x,n)` - Tipareste indicii marcati cu *true* (cercurile care pot fi eliminate).

Implementare

Varianta Pascal

```
Program Bile_Rostogolite;

Type    Vector = Array[0..10] Of Real;

Function Len(x:Vector):Byte;  Begin Len:=Trunc(x[0]) End;

Procedure Print(x:Vector);   Var i:Byte;
Begin
     Write(' X = '); For i:=1 To Len(x) Do Write(x[i]:8:5,', '); Writeln
End;

Function  Max(a,b:Real):Real;  Begin If a>b Then Max:=a Else Max:=b End;

Procedure Rost_B(r:Vector; Var x:Vector);          {   Det poz. xi a bilelor de raza ri  }
Var i,j:Byte;
Begin
     x[0]:=r[0]; x[1]:=r[1];
     For i:=2 To Len(r) Do   Begin
          x[i]:=r[i];                                          {  La zid       }
          For j:=1 To i-1 Do
              x[i]:=Max(x[i],x[j]+2*Sqrt(r[j]*r[i]));      {  Xmax daca s-ar lovi de toate bilele  }
     End
End;

Function Maxim(r,x:Vector; n: Byte): Real;
Begin
         If (n>0) Then Maxim:=Max(Maxim(r,x,n-1),x[n]+r[n])  Else Maxim:=0;
End;

Function Md(r,x:Vector): Real; Begin Md:=Maxim(r,x,Len(x)); End;

Const r:Vector = ( 7,   2, 3, 1, 7, 2, 10, 1  , 0,0,0 );
Var   x:Vector;

Begin
    Rost_B (r,x);
    Print  (x);
    Writeln(' Md = ',Md(r,x):8:5); Readln
End.
```

```
X =  2.00000,  6.89898, 10.36308, 16.06413, 23.54745, 32.79733, 39.12189,
Md = 42.79733
```

Varianta C++

```
 1  #include <iostream>
 2  #include <stdlib.h>
 3  #include <math.h>
 4
 5  using namespace std;
 6
 7  double d(double* r, int j, int i)                    /// d = Distanta (Cj,Ci) (xi-xj)
 8  {
 9         if (j) return 2*sqrt(r[j]*r[i]); else return r[i];  /// double Sqrt( double x,  double Eps) { ... }
10  }
11
12  double PozX(double* r, double* x, int j, int i)      /// Pozitia Cercului (bilei) i = Coord. x a centrului cercului i
13  {
14         return x[j]+d(r,j,i);
15  }
16
17  void Rost_B(double* r, double* x, bool* El)          /// det poz. xi a bilelor de raza ri
18  {
19       int  m=1;                                       /// m = nr. bilei care care determina marginea dreapta
20       x[1]=r[1],  El[1]=false;                        /// Prima bila se opreste in Oy
21       for (int i=2; i<=r[0]; i++)          {
22            int p=0; El[i]=false;                      /// La zid  (se loveste de Oy)
23            for (int j=1; j<=i-1; j++)
24                 if (PozX(r,x,j,i)>PozX(r,x,p,i)) p=j;
25            x[i]=PozX(r,x,p,i);
26            if (x[i]+r[i]>x[m]+r[m]) m=i;
27            if (p and p<i-1)
28                for (int j=p+1; j<=i-1; j++) El[j]=true;
29       }
30       for (int i=m+1; i<=r[0]; i++) El[i]=true;      /// Sunt marcate ultimele bile care pot fi eliminate
31  }                                                   ///         (cele care nu depasesc limita dreapta)
32
```

```
33  double Max(double* r, double* x, int n)                    /// Max{(xi+ri)/ i=1,n}
34  {
35     if (n) return Max(Max(r,x,n-1),x[n]+r[n]); else return 0;   ///  double Max(double x, double y) { if (x>y) return x; else return y;
36  }
37
38  double Md(double* r, double* x)                             /// Determina Marginea dreapta
39  {
40         return Max(r,x,r[0]);
41  }
42
43  void Print (bool* x, int n)                                 /// Tipareste indicii marcati cu true (ce poate fi eliminat)
44  {
45       if (n) {
46                 Print(x,n-1);
47                 if (x[n]) cout << n << ' ';
48  }             }
49
50  int main()
51  {
52       double r[] = {7,   2, 3, 1, 7, 2, 10, 1 }, x[int(r[0])+1];     /// n=r[0];
53       bool McEl[int(r[0])+1];                                        /// McEl = Multimea cercurilor care pot fi eliminate
54       Rost_B(r,x,McEl);
55       cout << "\n M.d. = " << Md(r,x);                              /// Md = Marginea dreapta
56       cout << "\n McEl = "; Print(McEl,r[0]);
57  }
```

Observaţie: Dacă cercurile ar fi rostogolite invers, spre marginea dreaptă, atunci vor avea poziţii finale diferite doar bilele care pot fi eliminate (3, 5, 7, aşa cum se poate vedea în figura alăturată). Dacă înainte 3 era tangent la 2, acum este la 4, 5 era la 4, acum este la 6, iar 7 era tangent la 6, acum este la *marginea dreaptă*. Această observaţie ne conduce spre o altă soluţie (o altă metodă de rezolvare a problemei) !

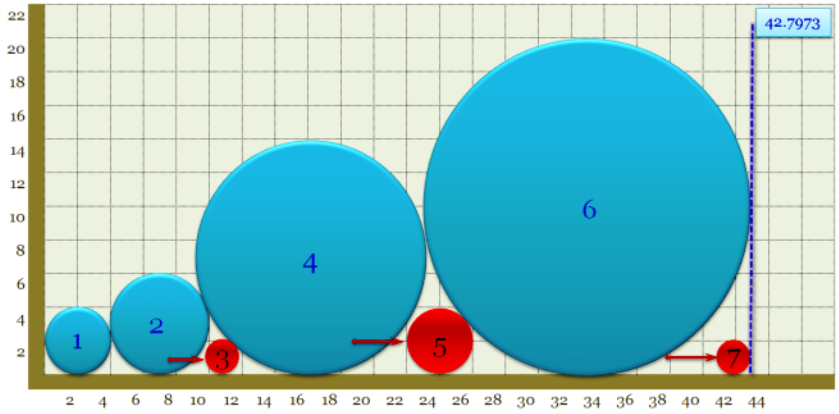

Fig. 4 Rostogolile spre dreapta

Probleme tip grilă

1. Fie șirul x=(5,3,2,1,1,1). Ce va realiza următorul algoritm?

```
Varianta Pascal
for i:=1 to n do
begin
      c:=x[i]
      x[i]:= x[n-i+1];
      x[n-i+1]:=c;
end;
for i:=1 to n do
      Write (x[i],' ');
```

```
Varianta C
for (i=1;i<=n;i++){
      c=x[i];
      x[i]=x[n-i+1];
      x[n-i+1]=c;
}
for (i=1;i<=n;i++)
      printf("%d", x[i]);
```

a) 1,1,2,1,3,5
b) 1,1,1,2,3,5
c) 5,3,2,1,1,1(CORECT)
d) Nici una dintre variantele anterioare nu este corecta.

2. Ce realizează următorul program:

```
Varianta Pascal
a:=1; b:=1;
for i:=2 to n do
begin
      if x[i]<x[a] then a:=i;
      if x[i]>x[b] then b:=i;
end;
writeln(a,' ',b);
```

```
Varianta C
a=b=1;
for(i=2;i<=n;i++){
      if (x[i]<x[a]) a=i;
      if (x[i]>x[b]) b=i;
}
printf("%d %d \n",a,b);
```

a) Afiseaza cele mai mari doua valori din sir.
b) Afiseaza cea mai mica si cea mai mare valoare din sir.
c) Afiseaza pozitiile celei mai mici si celei mai mari valori din sir.
d) Afiseaza pozitia primei aparitii a valorii minime si pozitia primei aparitii a valorii maxima din sir. (CORECT)

3. Ce realizează următorul program:

Varianta C

```
#include<iostream>
using namespace std;

double a (double* X, double& b, double& c) {     ///   a  …   b,  c,  = ???
    int fm=1, fM=fm;  double m=X[1], M=m;  b=c=1;
    for (int i=2; i<=X[0]; i++)
           if (X[i]< m)   fm=1,  m=X[i], b=i;  else
           if (X[i]==m)   b=(b*fm+i)/++fm;     else
           if (X[i]> M)   fM=1, M=X[i], c=i;   else
       if (X[i]==M)   c=(c*fM+i)/++fM;
    return (fm*m+fM*M)/(fm+fM) ;
}

int main(){
    double X[] = { 9,  3, 2, 4, 6, 2, 4, 3, 6, 3 },  b, c;
    cout << " a = " << a (X, b, c) << endl;
    cout << " b = " << b << ",   c = " << c << endl;
}
```

I. Care este semnificația variabilelor b și c?
II. Ce calculează (returnează) functia a?

a) Afiseaza cea mai mica si cea mai mare valoare din sir. (CORECT)
   I. b = media aritmetică a indicilor valorilor minime,
      c = media aritmetică a indicilor valorilor maxime;
   II. a = media aritmetică a valorilor sau (si) maxime.

b) Afiseaza cea mai mica si cea mai mare valoare din sir. (*Gresit*)
   I. b = media aritmetică a valorilor minime,
      c = media aritmetică a valorilor maxime;
   II. a = media aritmetică a valorilor minime si maxime.

Exemplu:

Pentru X = (3, 2, 4, 2, 2, 4, 3, 6, 3), rezultatul este:

i =	1	2	3	4	5	6	7	8	9

I. *b = Ma(ind. Min) = (2+4+5)/3 = 3.6666… (media aritm. a indicilor val. **minime**),*
*c = Ma(ind. Max) = 8 / 1 = 8 (media aritm. a indicilor val. **maxime**);*

II. *a = Ma(elem. Min si Max) = (2+2+2+6)/4 = 3 (media aritm. a val. min. sau max.).*

https://www.tutorialspoint.com/compile_pascal_online.php

https://www.tutorialspoint.com/compile_cpp_online.php