 **!!! ATENȚIE !!!** 

Aceste rezolvări NU au fost aprobate de MINISTERUL EDUCAȚIEI sau altă comisie recunoscută de Ministerul Educației. În consecință nimeni nu își asumă răspunderea pentru eventualele greşeli şi / sau perderi survenite în urma folosirii lor!

Foloseşte rezolvările pe riscul tău !!!

Dacă găseşti greşeli sau ai nelămuriri în legătură cu o anumită rezolvare trimite-mi un e-mail pe adresa raducu@trei.ro şi voi încerca să lămuresc / corectez problema.

Varianta 1:

1. b **2.** 6

3.
```
var f:text;
    n,nr:integer;
    gasit:boolean;
begin
  gasit:=false;
  assign(f,'bac.txt'); reset(f);
  write(' n= '); read(n);
  while not eof(f) do
    begin
      read(f,x);
      if x mod n = 0
         then  begin
                 write(x,' ');
                 gasit:=true;
               end;
    end;
  close(f);
  if gasit = false
    then write(' NU EXISTA ! ');
end.
```

4.
```
type sir=array[1..100]of integer;
var a:sir;
    n,i:integer;
    gasit:boolean;

function sub(v:sir;n,a:integer):integer;
var e,i:integer;
begin
  e:=0;
  for i:=1 to n do
    if v[i]=a
      then e:=e+1;
  sub:=e;
end;

begin
  write(' N= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  gasit:=false;
  i:=1;
  while (not gasit) and (i<n) do
    begin
```

```
        if sub(a,n,a[i])>1
          then gasit:=true;
        i:=i+1;
      end;
    if gasit
      then write(' NU')
      else write(' DA');
  end.
```

Varianta 2:

1. d **2.** 551100

3.
```
var a:array[1..100]of integer;
    f:text;   sortat:boolean;
    n, nr, i, aux:integer;
begin
  assign(f,'nr.txt');   reset(f);
  n:=0;
  while not eof(f) do
    begin
      read(f,nr);
      if nr >0
        then begin
               n:=n+1;
               a[n]:=nr;
             end;
    end;
  close(f);
  if n=0 then write(' NU EXISTA')
         else begin
                repeat
                  sortat:=true;
                  for i:=1 to n-1 do
                    if a[i]>a[i+1]
                      then begin
                             aux:=a[i];
                             a[i]:=a[i+1];
                             a[i+1]:=aux;
                             sortat:=false;
                           end;
                until sortat;
                for i:=1 to n do
                  write(a[i],' ');
              end;

end.
```

4.
```
var n,n1:integer;

function f(a:integer):integer;
var i,s,d:integer;
begin
```

```
    d:=2; s:=0;
    while a>1 do
      begin
        while a mod d=0 do
          begin
            a:=a div d;
            s:=s+1;
          end;
        d:=d+1;
      end;
    f:=s;
  end;
  begin
    write(' n= '); read(n);
    if n<10
      then n1:=2
      else n1:=n mod 10 *10 + n div 10;
    if (f(n)=1) and (f(n1)=1)
      then write(' DA ')
      else write(' NU ');
  end.
```

Varianta 3:

1. a **2.** xxxyyy

3.

```
var a:array[1..100]of integer;
    f:text;   sortat:boolean;
    n, nr, i, aux:integer;
begin
  assign(f,'nr.txt');   reset(f);
  n:=0;
  while not eof(f) do
    begin
      read(f,nr);
      if (nr > 99) or (nr < -99)
        then begin
               n:=n+1;
               a[n]:=nr;
             end;
    end;
  close(f);
  repeat
    sortat:=true;
    for i:=1 to n-1 do
      if a[i]>a[i+1]
        then begin
               aux:=a[i];
               a[i]:=a[i+1];
               a[i+1]:=aux;
```

```
                sortat:=false;
              end;
    until sortat;
    if n=0
      then write(' NU EXISTA! ')
      else for i:=1 to n do
             write(a[i],' ');
  end.
```

4.
```
  var n ,c, i, na:longint;


  function cif(a:longint; b:byte):byte;
  var na:byte;
  begin
    na:=0;
    while a>0 do
      begin
        if a mod 10 = b
          then na:=na+1;
        a:=a div 10;
      end;
    cif:=na;
  end;

  begin
    write(' n= '); read(n);
    c:=9;
    while c>0 do
      begin
        na:=cif(n,c);
        for i:=1 to na do write(c);
        c:=c-2;
      end;
  end.
```

Varianta 4:

1. c **2.** *4062

3.
```
  var a:array[1..100]of integer;
      f:text;   sortat:boolean;
      n, nr, i, aux:integer;
  begin
    assign(f,'nr.txt');   reset(f);
    n:=0;
    while not eof(f) do
      begin
        read(f,nr);
```

```
        if (nr < 100) and (nr > -100)
          then begin
                 n:=n+1;
                 a[n]:=nr;
               end;
      end;
    close(f);
    repeat
      sortat:=true;
      for i:=1 to n-1 do
        if a[i]<a[i+1]
          then begin
                 aux:=a[i];
                 a[i]:=a[i+1];
                 a[i+1]:=aux;
                 sortat:=false;
               end;
    until sortat;
    if n=0
      then write(' NU EXISTA! ')
      else for i:=1 to n do
             write(a[i],' ');
  end.
```

4.

```
var n ,c, i, na:longint;

function cif(a:longint; b:byte):byte;
var na:byte;
begin
  na:=0;
  while a>0 do
    begin
      if a mod 10 = b
        then na:=na+1;
      a:=a div 10;
    end;
  cif:=na;
end;

begin
  write(' n= '); read(n);
  c:=1;
  while c<10 do
    begin
      na:=cif(n,c);
      for i:=1 to na do write(c);
      c:=c+2;
    end;
end.
```

Varianta 5:

1. d **2.** 2****

3.
```
var n:longint;
    f:text;
begin
  assign(f,'nr.txt'); rewrite(f);
  write(' n= '); read(n);
  while n>0 do
    begin
      write(f,n,' ');
      n:=n div 10;
    end;
  close(f);
end.
```

4.
```
var n,i:integer;  gasit:boolean;
    a:array[1..100]of longint;
function f(a:longint):longint;
var d:longint;
begin
  d:=2;
  while a mod d <>0 do d:=d+1;
  f:=d
end;
begin
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  gasit:=false;
  for i:=1 to n do
    if f(a[i]) = a[i]
      then begin
             gasit:=true;
             write(a[i],' ');
           end;
  if not gasit
    then write(' NU EXISTA! ');
end.
```

Varianta 6:

1. a **2.** 143

3.
```
type sir=array[0..100]of integer;
```

```
var n,i,i1,i2:integer;
    a:sir;
    f:text;
procedure s1(var a,b:integer);
begin
  a:=a+b;
  b:=a-b;
  a:=a-b;
end;
function s2(a:sir; p:integer;q:integer):integer;
var i:integer;
begin
  if p<q then begin
                i:=p;
                while (i<=q) and (a[i] mod 5 <> 0) do
                  i:=i+1;
                if i = q+1
                  then s2:=-1
                  else s2:=i;
              end
         else begin
                i:=p;
                while (i>=q) and (a[i] mod 5 <> 0) do
                  i:=i-1;
                if i = q-1
                  then s2:=-1
                  else s2:=i;
              end
end;
begin
  assign(f,'bac.txt');  rewrite(f);
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  i1:=s2(a,1,n);
  i2:=s2(a,n,1);
  s1(a[i1],a[i2]);
  writeln(i1,' ',i2);
  for i:=1 to n do
    write(f,a[i],' ');
  close(f);
end.
```

Varianta 7:

1. d **2.** 126

3.
```
var a:array[1..300] of integer;
    n,i,aux:integer;
begin
  write(' n= '); read(n);
  for i:=1 to 3*n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  for i:=1 to n do
    begin
      aux:=a[i];
      a[i]:=a[2*n+i];
      a[2*n+i]:=aux;
    end;
  for i:=1 to 3*n do
    write(a[i],' ');
end.
```

4.
```
var s,n:longint;
    g:text;

function f(n:longint):longint;
var x,y:longint;
begin
  x:=1;
  while x<n do
    begin
      if x<5
        then x:=x+1
        else x:=2*x;
    end;
  if n>5
    then f:=x div 2
    else f:=x;
end;

begin
  assign(g,'numere.txt'); rewrite(g);
  write(' s= '); read(s);
  while s>1 do
    begin
      n:=f(s);
      write(g,n,' ');
      s:=s-n;
    end;
  close(g);
end.
```

Varianta 8:

1. c **2.** 4 numere (105, 123, 321, 501)

3.
```
var f:text;
    i,j,n:integer;
begin
  assign(f,'bac.txt'); rewrite(f);
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      for j:=0 to n-i do
        write(f,2*j,' ');
      writeln(f);
    end;
  close(f);
end.
```

4.
```
var n:longint;
    c0,s:integer;
function p1(n:longint):integer;
var s:integer;
begin
  s:=0;
  while n>0 do
    begin
      s:=s+n mod 10;
      n:=n div 10;
    end;
  p1:=s;
end;
function p2(n:longint):longint;
begin
  p2:=n div 10;
end;
begin
  write(' n= '); read(n);
  s:=p1(n); n:=p2(n);
  while n>0 do
    begin
      if s=p1(n)
        then c0:=c0+1;
      s:=p1(n);
      n:=p2(n);
    end;
  write(' c0= ',c0);
end.
```

Varianta 9:

1. a **2.** 167

3.
```
type sir=array[0..100]of integer;
var n,i,j:integer;
    a:sir;
    f:text;

procedure s1(a:sir; var p, q:integer);
var i:integer;
begin
  i:=p;
  while (i<=q) and (a[i] mod 2 <> 0) do i:=i+1;
  if i = q+1
     then p:=-1
     else p:=i;

  i:=q;
  while (i>=p) and (a[i] mod 2 = 0) do i:=i-1;
  if i = q-1
     then q:=-1
     else q:=i;
end;

procedure s2(var a,b:integer);
begin
  a:=a+b;
  b:=a-b;
  a:=a-b;
end;

begin
  assign(f,'bac.txt');  rewrite(f);
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  i:=1; j:=n;
  s1(a,i,j);
  while (i<>-1) and (j<>-1) and (i<j) do
    begin
      s2(a[i],a[j]);
      s1(a,i,j);
    end;
  for i:=1 to n do
    write(f,a[i],' ');
  close(f);
end.
```

Varianta 10:

1. b **2.** 56

3.
```
var a:array[1..100] of integer;
    n,i,j,s:integer;
begin
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  for i:=1 to n do
    begin
      s:=0;
      for j:=1 to n-i+1 do
        s:=s+a[j];
      writeln(s);
    end;
end.
```

4.a)
```
var i,n,nr,p2:longint;
    f:text;
begin
  assign(f,'bac.txt'); rewrite(f);
  write(' n= '); read(n);
  p2:=1;
  for i:=1 to n do
    begin
      write(' nr= '); read(nr);
      while p2<nr do p2:=p2*2;
      write(f,p2,' ');
    end;
  close(f);
end.
```

4.b) În variabila p2 reţin puterile lui 2. Initial este 2^0 adica 1. Citesc, pe rând, cele n numere. Dacă numarul citit este mai mare decât p2 aflu care este puterea lui 2 astfel încât să fie respectate condiţiile, apoi afişez valoarea determinată.

Programul este eficient deoarece nu foloseşte structuri de date pentru a reţine toate elementele şirului şi nu determina de fiecare dată 2^x ci numai adaugă la ultima putere, (dacă este necesar). În concluzie programul consumă puţină memorie şi este rapid.

Varianta 11:

1. b **2.** 8

3.
```
type sir=array[1..100]of integer;
var a,b:sir;
    n,i,m:integer;

procedure P(a:sir; k:integer; var max:integer);
var i:integer;
begin
  max:=a[1];
  for i:=1 to k do
    if max<a[i]
      then max:=a[i];
end;

begin
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  for i:=1 to n do
    begin
      P(a,i,m);
      b[i]:=m;
    end;
  for i:=1 to n do
    write(b[i],' ');
end.
```

4.
```
var a:array[1..100]of longint;
    i,n,np:longint;

function prim(x:longint):boolean;
var d:longint;
begin
  d:=2;
  while (x mod d <>0)and (x>1) do d:=d+1;
  if d=x
    then prim:=true
    else prim:=false;
end;

begin
  np:=0;
  write(' N= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= ');
      read(a[i]);
    end;
  for i:=1 to n do
    if prim(a[i])
```

```
        then np:=np+1;
    write(' np= ', np);
  end.
```

Varianta 12:

1. c **2.** 2+2+2+3; 2+2+5; 2+7

3.a) Citesc primele m numere într-un şir apoi următoarele n numere pe rând. După citirea unui nou număr continu parcurgerea şirului până ajung la o valoare care nu este mai mică decat cea citită. Fiecare valoare din sir peste care sar este numarată. Dacă ultima valoare parcursă în sir este diferită de cea citită atunci am mai descoperit o valoare care apare o singura dată.

Programul este eficient deoarece nu reține toate elementele deci consumă puțnnă memorie şi percurge şirul o singură data deci este rapid.

3.b)

```
var m,n,x,i,j:integer;
    a:array[1..101]of longint;
    nr:longint;
begin
 write(' m= '); read(m);
 write(' n= '); read(n);
 for i:=1 to m do
   begin
     write(' a[',i,']= ');
     read(a[i]);
   end;
 x:=0; j:=1;
 for i:=1 to n do
   begin
     write(' nr= ');
     read(nr);
     while (a[j]<nr) and (j<=m) do begin
                                     x:=x+1;
                                     j:=j+1;
                                   end;
     if a[j]<>nr then x:=x+1
                 else j:=j+1;
   end;
 write(' Numere distincte: ',x);
end.
```

4.

```
var n,i,cd:longint;
function nr(n:longint; c:byte): byte;
var x:byte;
begin
  x:=0;
  while n >0 do
    begin
```

```
        if n mod 10 = c
          then x:=x+1;
        n:=n div 10;
      end;
    nr:=x;
  end;

  begin
    write(' n= '); read(n);
    cd:=0;
    for i:=0 to 9 do
      if nr(n,i)>0
        then cd:=cd+1;
    write(' cifre dist: ',cd);
  end.
```

Varianta 13:

1. d **2.** 6

3.
```
var i,j,n:integer;
begin
  write(' n= '); read(n);
  i:=1;
  while n>0 do
    begin
      j:=i;
      while (j>0) and (n>0) do
        begin
          write(j,' ');
          j:=j-1;
          n:=n-1;
        end;
      i:=i+1;
    end;
end.
```

4.
```
var f,g:text;
    nr:longint;

procedure P(var n:longint; c:byte);
var i:integer;
    s,s2:string[10];
begin
  str(n,s); str(c,s2);
  for i:=1 to length(s) do
    if s[i]=s2
      then delete(s,i,1);
  val(s,n,i);
end;
```

```
begin
  assign(f,'bac.in'); reset(f);
  assign(g,'bac.out'); rewrite(g);
  while not eof(f) do
    begin
      read(f,nr);
      P(nr,1);
      P(nr,3);
      P(nr,5);
      P(nr,7);
      P(nr,9);
      if nr>0 then write(g,nr,' ');
    end;
  close(f);
  close(g);
end.
```

Varianta 14:

1. b **2.** 72

3.
```
var f:text;
    nr,c:longint;
begin
  assign(f,'bac.txt'); reset(f);
  c:=0;
  while not eof(f) do
    begin
      read(f,nr);
      write(nr,' ');
      c:=c+1;
      if c mod 5 =0
        then writeln;
    end;
  close(f);
end.
```

4.
```
var i,ii:integer;
function prim(n:integer):byte;
var d:integer;
begin
  d:=2;
  while n mod d<>0 do d:=d+1;
  if n=d then prim:=1
         else prim:=0;
end;
begin
  for i:=11 to 97 do
```

```
      begin
        ii:=i mod 10 * 10 + i div 10;
        if (prim(i)=1) and (prim(ii)=1)
          then write(i,' ');
      end;
  end.
```

Varianta 15:

1. b **2.** 85

3.
```
var n,n5:longint;
begin
  n5:=0;
  write(' n= '); read(n);
  while n<>0 do
    begin
      while n>0 do
        begin
          if n mod 10 =5 then n5:=n5+1;
          n:=n div 10;
        end;
      write(' n= '); read(n);
    end;
  write(' n5= ',n5);
end.
```

4.a. Consieer inițial ultimul nr impar -1. Citesc pe rând cate un numar si verific dacă este impar. În caz afirmativ îl reţin. La final dacă valoarea din ni este diferita de -1 înseamnă ca fişierul conţinea cel puţin un număr impar si afişez valoarea variablei altfel textul cerut.

Algritmul este eficient fiindcă prelucrează o singură dată numerele din fişier şi nu foloseşte structuri de date (şiruri) pentru a rține elementele

4.b.
```
var f:text;
    nr,ni:longint;
begin
  ni:=-1;
  assign(f,'bac.in'); reset(f);
  while not eof(f) do
    begin
      read(f,nr);
      if nr mod 2 <> 0
        then ni:=nr;
    end;
  if ni=-1
    then write(' Nu exista numere impare ')
    else write(ni);
end.
```

Varianta 16:

1. d **2.** 77755, 77757, 77777

3.
```
var a:array[1..10] of integer;
    i,m13:integer;
begin
  write(' Numerele: ');
  for i:=1 to 10 do read(a[i]);
  m13:=0;
  for i:=1 to 10 do
    if a[i] mod 13=0
      then m13:=m13+1;
  write(m13,' ');
  for i:=1 to 10 do
    if a[i] mod 13 = 0
      then write(i,' ');
end.
```

4.
```
var f:text;
    n:longint;

function cifrak(n:longint; k:byte):byte;
var nc:byte;
begin
  nc:=0;
  while n>0 do
    begin
      if n mod 10 = k
        then nc:=nc+1;
      n:=n div 10;
    end;
  cifrak:=nc;
end;

begin
  assign(f,'numere.txt'); reset(f);
  while not eof(f) do
    begin
      read(f,n);
      if cifrak(n,0)=3
        then write(n,' ');
    end;
  close(f);
end.
```

Varianta 17:

---

1. c **2.** 12347, 12346, 12345

3.
```
function interval(v:sir; d:byte):byte;
var i,ne:byte;
begin
  if v[1]>v[n]
    then begin
           ne:=v[1];
           v[1]:=v[n];
           v[n]:=ne;
         end;
  ne:=0;
  for i:=1 to n do
    if (v[i]>=v[1]) and (v[i]<=v[n])
      then ne:=ne+1;
  interval:=ne;
end;
```

4.a. Pun pe prima pozitie din sir valoarea 0 apoi citesc elementele în şir începand cu adoua poziție. La final alelez funcția interval si afişez valoarea returnată - 1

4.b.
```
type sir=array[1..100]of integer;
var f:text;
    a:sir;
    n,i:integer;
function interval(v:sir; d:byte):byte;
var i,ne:byte;
begin
  if v[1]>v[d]
    then begin
           ne:=v[1];
           v[1]:=v[d];
           v[d]:=ne;
         end;
  ne:=0;
  for i:=1 to d do
    if (v[i]>=v[1]) and (v[i]<=v[d])
      then ne:=ne+1;
  interval:=ne;
end;

begin
  assign(f,'numere.txt'); reset(f);
  a[1]:=0;
  i:=2;
  while not eof(f) do
    begin
      read(f,a[i]);
      i:=i+1;
    end;
```

```
    n:=i-1;
    write(interval(a,n)-1);
    close(f);
  end.
```

Varianta 18:

1. b **2.** 11101, 11110, 11111

3.
```
function count(v:sir; n:byte):byte;
var i,ne:integer;
begin
  ne:=0;
  for i:=1 to n do
    if v[i]>=(v[1]+v[n])/2
      then ne:=ne+1;
  count:=ne;
end;
```

4.a. Pun pe prima pozitie din sir valoarea 0 apoi citesc elementele în şir începand cu adoua poziție. La final alelez funcția count şi afişez valoarea returnată

4.b.
```
type sir=array[1..100]of integer;
var a:sir;
    n,i:integer;
    f:text;

function count(v:sir; n:byte):byte;
var i,ne:integer;
begin
  ne:=0;
  for i:=1 to n do
    if v[i]>=(v[1]+v[n])/2
      then ne:=ne+1;
  count:=ne;
end;

begin
  assign(f,'numere.txt'); reset(f);
  a[1]:=0;
  i:=2;
  while not eof(f) do
    begin
      read(f,a[i]);
      i:=i+1;
    end;
  n:=i-1;
  write(count(a,n));
  close(f);
end.
```

Varianta 19:

1. a **2.** 10349, 10352, 10354

3.
```
type sir=array[1..100]of real;
var a:sir;
    n,i:integer;

procedure aranjare(var v:sir; n:byte);
var i,j:byte; aux:real;
begin
  i:=1; j:=n;
  while i<j do
    begin
      while (i<j) and (v[i]<0) do i:=i+1;
      while (i<j) and (v[j]>0) do j:=j-1;
      if i<j
        then begin
               aux:=v[i];
               v[i]:=v[j];
               v[j]:=aux;
             end;
    end;
end;

begin
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= '); read(a[i]);
    end;
  aranjare(a,n);
  for i:=1 to n do
    write(a[i]:7:2,' ');
end.
```

4.a. Citesc cele 2 numere apoi citesc, pe rand, cate o linie din fisier şi o prelucrez cu ajutorul subprogramului. Linia prelucrata o scriu in fisierul indicat.

4.b.
```
type sir=array[1..100]of real;
var f,g:text;
    n,m,i,j:integer;
    a:sir;

procedure aranjare(var v:sir; n:byte);
var i,j:byte; aux:real;
begin
  i:=1; j:=n;
  while i<j do
    begin
      while (i<j) and (v[i]<0) do i:=i+1;
      while (i<j) and (v[j]>0) do j:=j-1;
```

```
        if i<j
          then begin
                 aux:=v[i];
                 v[i]:=v[j];
                 v[j]:=aux;
               end;
     end;
  end;

  begin
    assign(f,'nr1.txt'); reset(f);
    assign(g,'nr2.txt'); rewrite(g);
    readln(f,n,m);
    for i:=1 to n do
      begin
        for j:=1 to m do
          read(f,a[j]);
        aranjare(a,m);
        for j:=m downto 1 do
          write(g,a[j]:7:2,' ');
        writeln(g);
      end;
    close(f);
    close(g);
  end.
```

Varianta 20:

1. c **2.** 35789, 35679, 35678

3.

```
type sir=array[1..100]of real;
var a:sir;
    n,i:integer;

procedure nule(var v:sir; n:byte);
var i,j:byte; aux:real;
begin
  i:=1; j:=n;
  while i<j do
    begin
      while (i<j) and (v[i]<>0) do i:=i+1;
      while (i<j) and (v[j]=0) do j:=j-1;
      if i<j
        then begin
               aux:=v[i];
               v[i]:=v[j];
               v[j]:=aux;
             end;
    end;
end;
```

```
begin
  write(' n= '); read(n);
  for i:=1 to n do
    begin
      write(' A[',i,']= '); read(a[i]);
    end;
  nule(a,n);
  for i:=1 to n do
    write(a[i]:7:2,' ');
end.
```

4.a. Citesc cele 2 numere apoi citesc, pe rand, cate o linie din fisier şi o prelucrez cu ajutorul subprogramului. Linia prelucrata o scriu in fisierul indicat.

4.b.
```
type sir=array[1..100]of real;
var f,g:text;
    n,m,i,j:integer;
    a:sir;

procedure nule(var v:sir; n:byte);
var i,j:byte; aux:real;
begin
  i:=1; j:=n;
  while i<j do
    begin
      while (i<j) and (v[i]<>0) do i:=i+1;
      while (i<j) and (v[j]=0) do j:=j-1;
      if i<j
        then begin
               aux:=v[i];
               v[i]:=v[j];
               v[j]:=aux;
             end;
    end;
end;

begin
  assign(f,'nr1.txt'); reset(f);
  assign(g,'nr2.txt'); rewrite(g);
  readln(f,n,m);
  for i:=1 to n do
    begin
      for j:=1 to m do
        read(f,a[j]);
      nule(a,m);
      for j:=m downto 1 do
        write(g,a[j]:7:2,' ');
      writeln(g);
    end;
  close(f);
  close(g);
end.
```

Varianta 21:

1. c **2.** `#######`

3.
```
function i_prim(n:integer):integer;
var p1,p2,i:integer;
    prim:boolean;
begin
  p1:=n+1;
  repeat
    p1:=p1-1;
    prim:=true;
    for i:=2 to p1 div 2 do
      if p1 mod i = 0
        then prim:=false;
  until prim;
  p2:=n-1;
  repeat
    p2:=p2+1;
    prim:=true;
    for i:=2 to p2 div 2 do
      if p2 mod i = 0
        then prim:=false;
  until prim;
  i_prim:=p2-p1;
end;
```

4.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 22:

1. a **2.** `#######`

3.
```
var k,n:integer;
function nz(n:integer):integer;
var nr2,nr5,x,i:integer;
begin
  nr2:=0;  nr5:=0;
  for i:=2 to n do
    begin
```

```
        x:=i;
        while x mod 2=0 do
          begin
            nr2:=nr2+1;
            x:= x div 2;
          end;
        while x mod 5=0 do
          begin
            nr5:=nr5+1;
            x:= x div 5;
          end;
      end;
    if nr2<nr5
      then nz:=nr2
      else nz:=nr5 ;
  end;

  begin
    write('k= '); read(k);
    n:=1;
    while nz(n)<k do n:=n+1;
    write(n,'!');
  end.
```

4.
```
var a,b,i,n,p:integer;
    f:text;

begin
  assign(f,'bac.txt');  reset(f);
  read(f,n);
  for i:=1 to n do
    begin
      readln(f,a,b);
      p:=2;
      while p<b do p:=p*2;
      if p div 2 >=a
        then write(p div 2,' ')
        else write(0,' ');
    end;
  close(f);
end.
```

Varianta 23:

1. d **2.**

3.
```
type sir=array[1..100]of integer;
var n,i:integer;
    x:sir;

procedure shift(var a:sir;n:integer);
```

```
var aux,i:integer;
begin
  aux:=a[1];
  for i:=1 to n-1 do
    a[i]:=a[i+1];
  a[n]:=aux;
end;

begin
  write(' n= '); read(n);
  write(' Elementele lui x sunt....');
  for i:=1 to n do
     read(x[i]);
  for i:=n downto 2 do
    shift(x,i);
  for i:=1 to n do
    write(x[i],' ');
end.
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 24:

1. a **2.** f(17)=3 f(22)=2

3.

```
type sir=array[1..100]of integer;
var x:sir;           M:real;
    n,i,min,max,sum:integer;

procedure P(n:integer; x:sir; var
mini,maxi,sum:integer);
begin
  mini:=x[1];
  for i:=1 to n do
    if x[i]<mini
      then mini:=x[i];
  maxi:=x[1];
  for i:=2 to n do
    if maxi<x[i]
      then maxi:=x[i];
  sum:=0;
  for i:=1 to n do
    sum:=sum+x[i];
```

```
end;
begin
  write(' n= '); read(n);
  write(' Introdu alorile: ');
  for i:=1 to n do
    read(x[i]);
  P(n,x,min,max,sum);
  M:=(sum-min-max)/(n-2);
  write(M:0:3);
end.
```

4.a.
```
var v:array[1..30000]of integer;
    n,a,b,i,min:integer;
    f:text;
begin
  assign(f,'bac.txt');   reset(f);
  read(f,n);
  for i:=1 to n do
    read(f,v[i]);
  read(f,a,b);
  min:=b+1;
  for i:=1 to n do
    if (v[i]<min) and (v[i]>=a)
      then min:=v[i];
  if min>b
    then write(' NU ')
    else write(min);
  close(f);
end.
```

4.b.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 25:

1. d **2.** a. f(16)=0 b. 95

3.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.a.
```
var f:text;
    x:real;
    p,n1,n2:longint;
begin
  assign(f,'numar.txt');   reset(f);
  read(f,x); p:=1;
  while x<>trunc(x) do begin
                         x:=x*10;
                         p:=p*10;
                       end;
  n1:=trunc(x);
  n2:=p;
  while n1<>n2 do
    if n1>n2
      then n1:=n1-n2
      else n2:=n2-n1;
  write(trunc(x) div n1,' ',p div n2);
  close(f);
end.
```

4.b.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 26:

1. b **2.**

3.
```
var i,k,n:integer;
begin
  write(' n= '); read(n);
  write(' k= '); read(k);
  for i:=k downto 1 do
    write(n*i,' ');
end.
```

4.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
```

```
##################################################
##################################################
```

Varianta 27:

1. c **2.** 17263544444

3.
```
function nreal(x,y:integer):real;
var y1:real;
begin
  y1:=y;
  while y1>1 do
    y1:= y1 /10;
  nreal:=x+y1;
end;
```

4.a.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.b.
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 28:

1. a **2.** #########

3.
```
var a:array[1..10000]of real;
    n,i,c:integer;
    s:real;
begin
  write(' N= '); read(n);
  for i:=1 to n do
    begin
```

```
        write(' A[',i,']=');
        read(a[i]);
      end;
    s:=0;
    for i:=1 to n do
      s:=s+a[i];
    c:=0;
    for i:=1 to n do
      if a[i] = (s-a[i]) / (n-1)
        then c:=c+1;
    write(' c= ',c);
  end.
```

4.

```
var a:array[1..10000]of integer;
    n,i,d1,d2,map:integer;
    gasit:boolean;
    f:text;

function primul(a:integer):integer;
var d:integer;
begin
  d:=2;
  while a mod d <> 0 do d:=d+1;
  primul:=d;
end;

begin
  assign(f,'numere.in'); reset(f);
  readln(f,n);
  for i:=1 to n do
    read(f,a[i]);
  gasit:=false;  map:=0;
  for i:=1 to n do
    begin
      d1:=primul(a[i]);
      d2:=a[i] div d1;
      if (primul(d1)=d1) and (primul(d2)=d2) and (a[i]>map)
        then begin
               gasit:=true;
               map:=a[i];
             end;
    end;
  if gasit
    then write(' DA ',map)
    else write(' NU ');
end.
```

Varianta 29:

1. c **2.** 1231210123

3.
```
function multiplii(a,b,c:integer):integer;
var i, m:integer;
begin
  m:=0;
  for i:=a to b do
    if i mod c = 0
      then m:=m+1;
  multiplii:=m
end;
```

4.
```
var a,b:array[1..100]of integer;
    n,m,i,j,s:integer;
    ok:boolean;    f:text;
begin
  assign(f,'numere.in'); reset(f);
  readln(f,n,m);
  for i:=1 to n do
    read(f,a[i]);
  for i:=1 to m do
    read(f,b[i]);
  i:=1; j:=1;
  ok:=true;
  while (j<=m) and (ok) do
    begin
      s:=0;
      while (s<b[j]) and (i<=n) do
        begin
          s:=s+a[i];
          i:=i+1;
        end;
      if s<>b[j]
        then ok:=false;
      j:=j+1;
    end;
  if ok
    then write(' DA ')
    else write(' NU ');
end.
```

Varianta 30:

1. a **2.** ###########

3.
```
type sir=array[1..100]of integer;

function suma(v:sir; n, i, j:integer):integer;
var s,i1:integer;
begin
```

```
    s:=0;
    for i1:=1 to i-1 do
      s:=s+v[i1];
    for i1:=j+1 to n do
      s:=s+v[i1];
    suma:=s;
  end;
```

4.a.
```
var f:text;
    a:real;
    n,i,c,upi:integer;
begin
  assign(f,'numere.in'); reset(f);
  readln(f,n);
  c:=0; upi:=0;
  for i:=1 to n do
    begin
      read(f,a);
      if trunc(a)>upi
        then begin
               upi:=trunc(a);
               c:=c+1;
             end;
    end;
  write(' numar de intervale= ', c);
end.
```

4.b. Ma bazez pe faptul ca numerele sunt ordonate crescator si dupa ce citesc un numar determin imediat din ce interval face parte. Daca este vorba despre un interval nou il numar.

Programul este eficient din punct de vedere al timpului de executare deoarece numerele se citesc o singura data iar imediat după citire acestea sunt si prelucrate. Programul este eficient din punct de vedere al memoriei utilizate deoarece nu retine toate numerele ci numai cate unu, imediat după ctire facand si prelucrarea valorii.

Varianta 31:

1. **2.**

3.

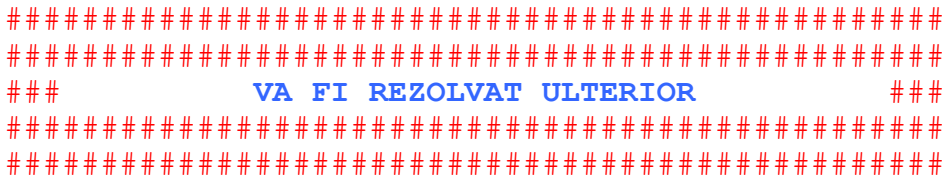
```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

Varianta 32:

1. **2.**

3.



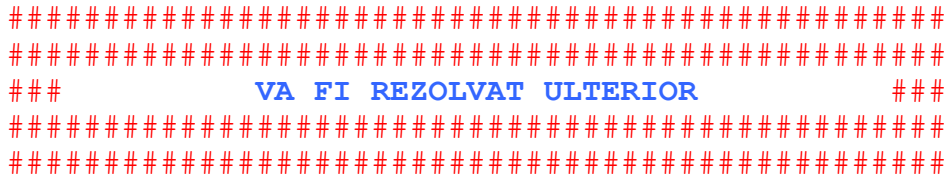
4.

Varianta 33:

1. **2.**

3.

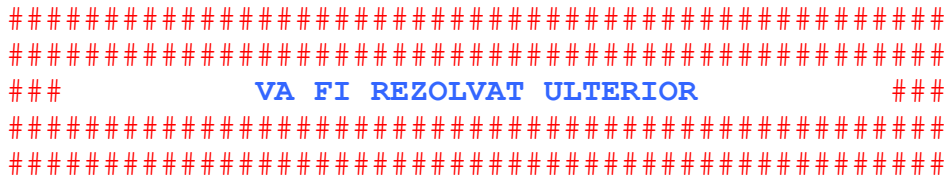
4.



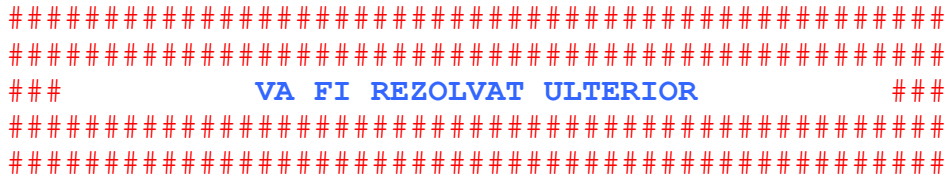
Varianta 34:

1. **2.**

3.



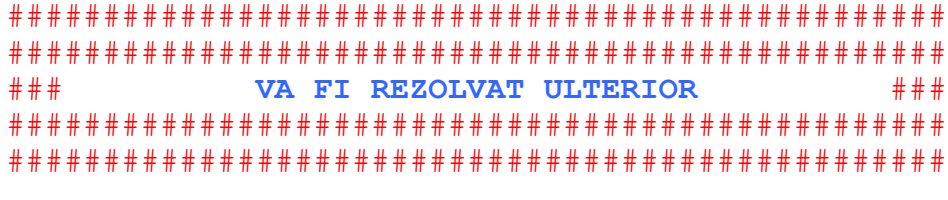
4.



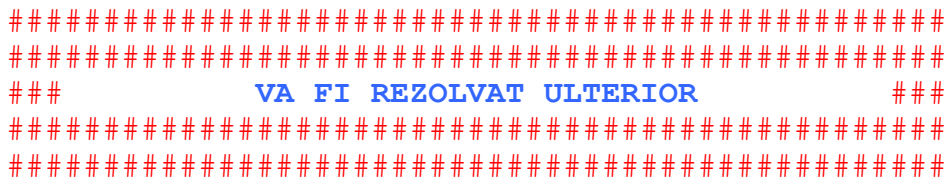
Varianta 35:

1. **2.**

3.



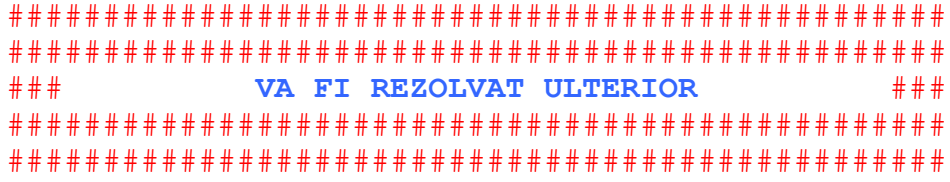
4.



Varianta 36:

1. **2.**

3.

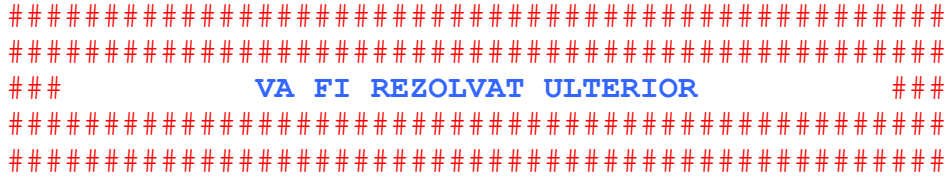


4.

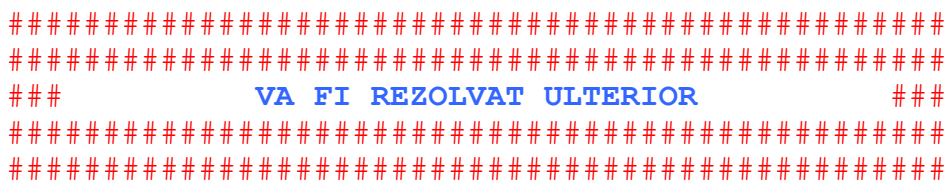
Varianta 37:

1. **2.**

3.



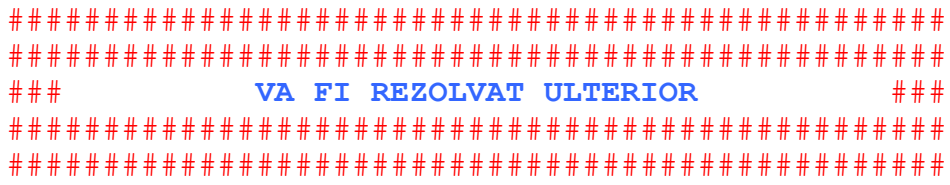
4.



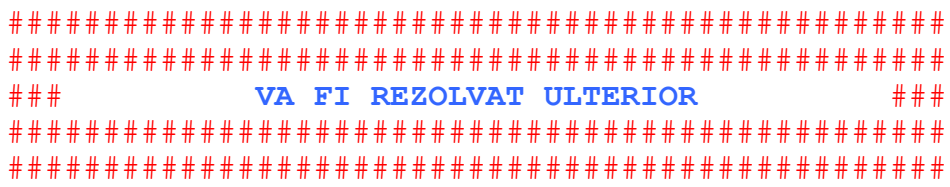
Varianta 38:

1. **2.**

3.



4.



Varianta 39:

1. **2.**

3.

4.

```
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 40:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 41:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
```

```
##################################################
```

Varianta 42:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

Varianta 43:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

Varianta 44:

1. **2.**

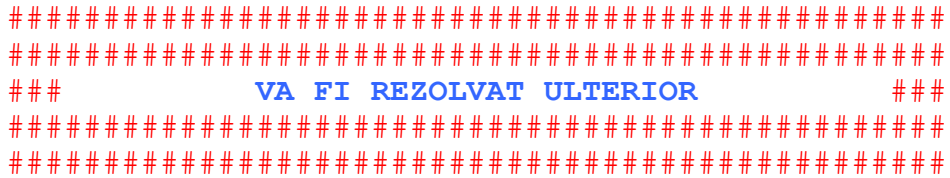
3.

4.

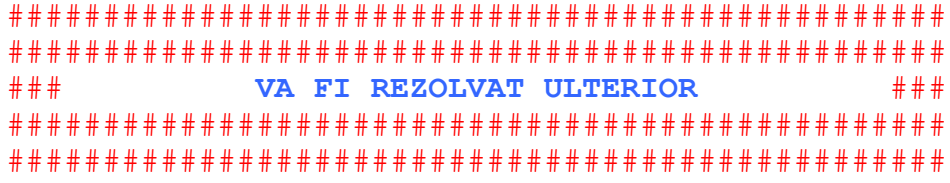
Varianta 45:

1. **2.**

3.



4.



Varianta 46:

1. **2.**

3.

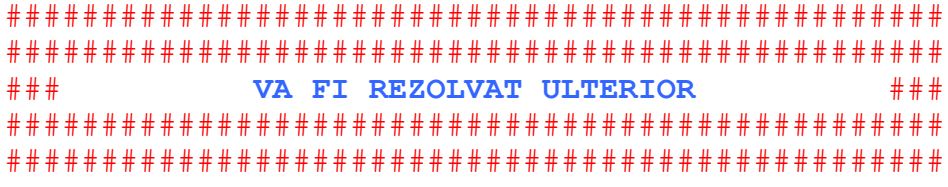
4.

Varianta 47:

1. **2.**

3.

4.



Varianta 48:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 49:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

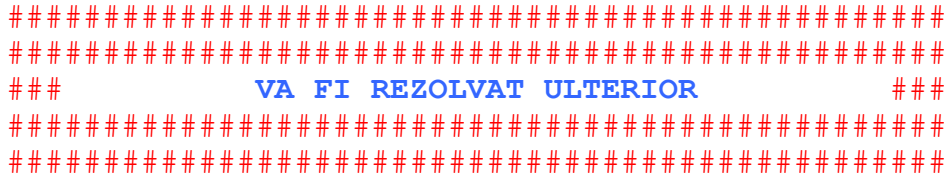
4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 50:

1. **2.**

3.



4.

Varianta 51:

1. **2.**

3.

4.

Varianta 52:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 53:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
```

```
##################################################
```

Varianta 54:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

Varianta 55:

1. **2.**

3.

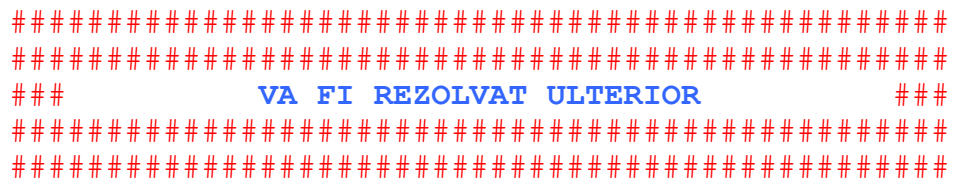
```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

4.

Varianta 56:

1. **2.**

3.

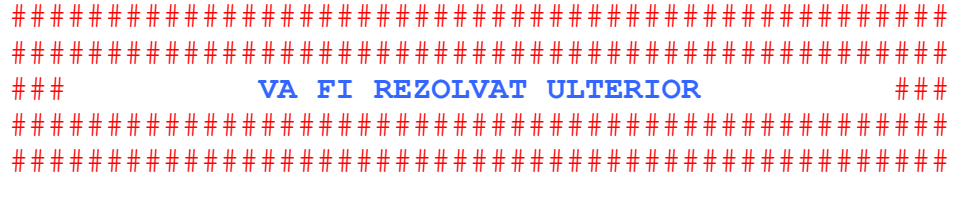


4.

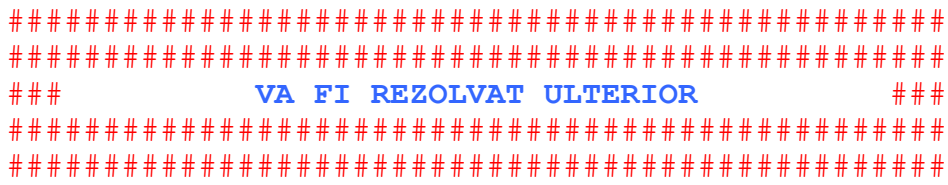
Varianta 57:

1. **2.**

3.



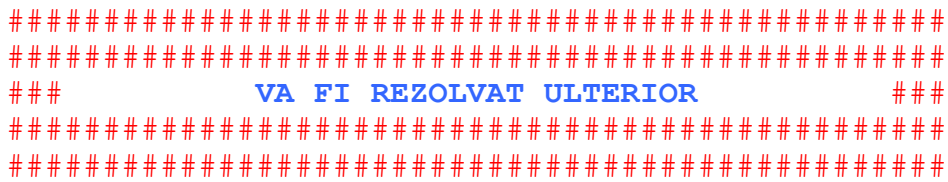
4.



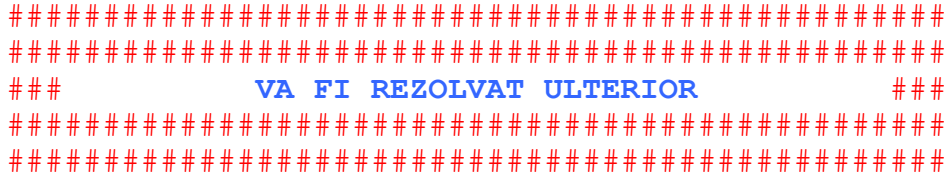
Varianta 58:

1. **2.**

3.



4.



Varianta 59:

1. **2.**

3.

```
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 60:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 61:

1. **2.**

3.

4.

Varianta 62:

1. **2.**

3.

4.

Varianta 63:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 64:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 65:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 66:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
```

```
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 67:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

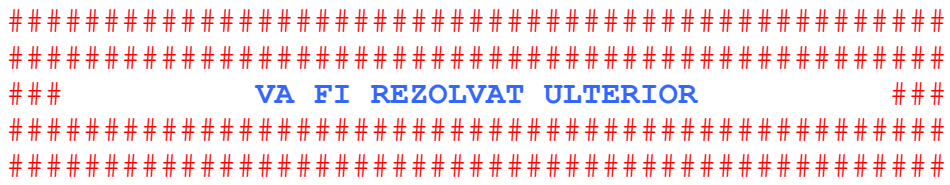
Varianta 68:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

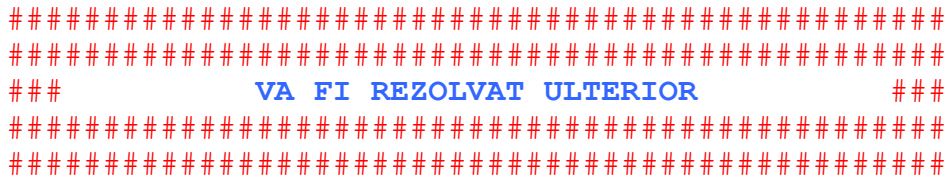
4.



Varianta 69:

1. **2.**

3.



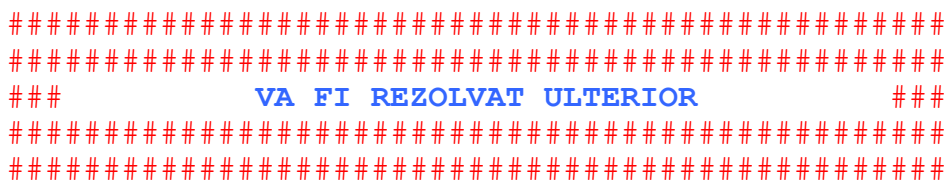
4.

Varianta 70:

1. **2.**

3.

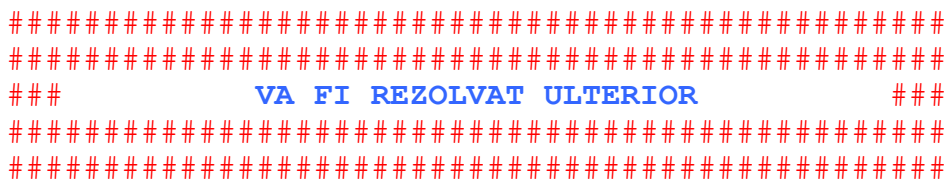
4.



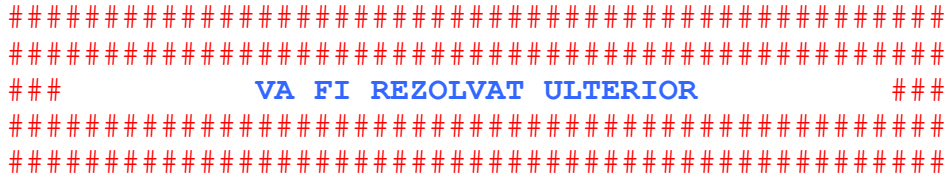
Varianta 71:

1. **2.**

3.



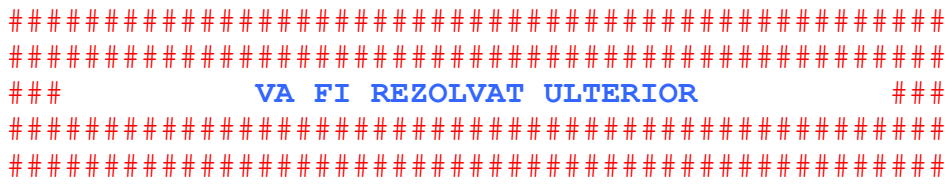
4.



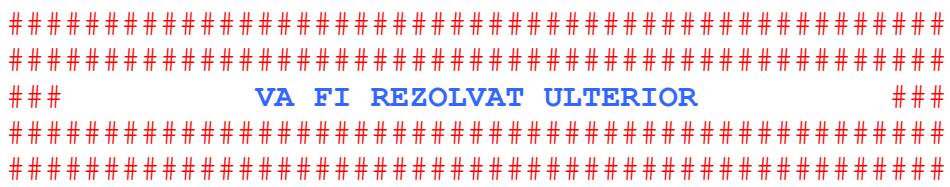
Varianta 72:

1. **2.**

3.



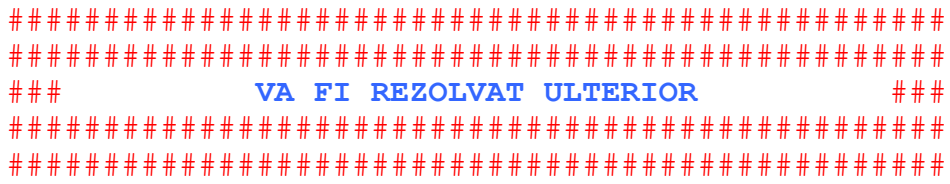
4.



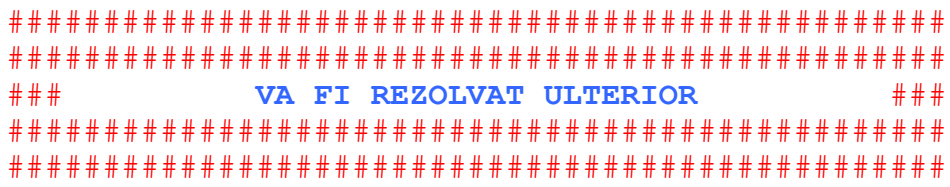
Varianta 73:

1. **2.**

3.



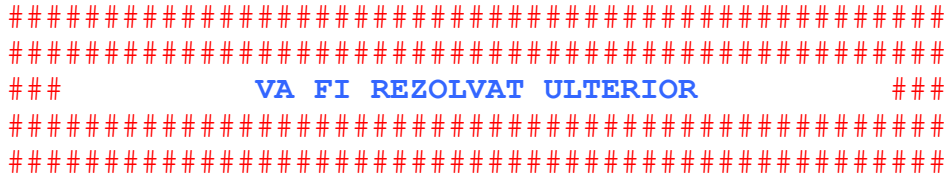
4.



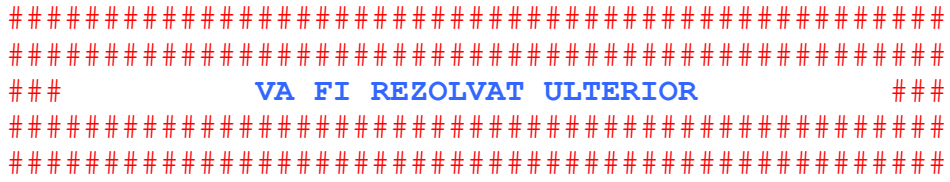
Varianta 74:

1. **2.**

3.



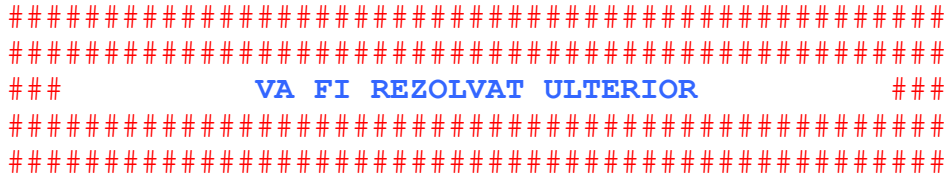
4.



Varianta 75:

1. **2.**

3.



4.

Varianta 76:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR             ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR             ###
##################################################
##################################################
```

Varianta 77:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR             ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR             ###
##################################################
```

```
################################################
```

Varianta 78:

1. **2.**

3.

```
################################################
################################################
###           VA FI REZOLVAT ULTERIOR          ###
################################################
################################################
```

4.

```
################################################
################################################
###           VA FI REZOLVAT ULTERIOR          ###
################################################
################################################
```

Varianta 79:

1. **2.**

3.

```
################################################
################################################
###           VA FI REZOLVAT ULTERIOR          ###
################################################
################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 80:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

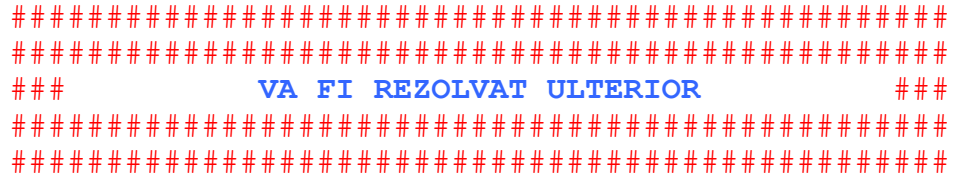
Varianta 81:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

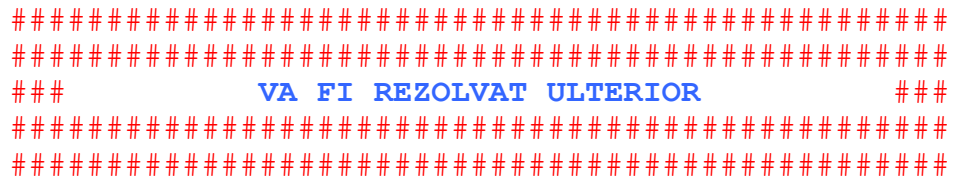
4.



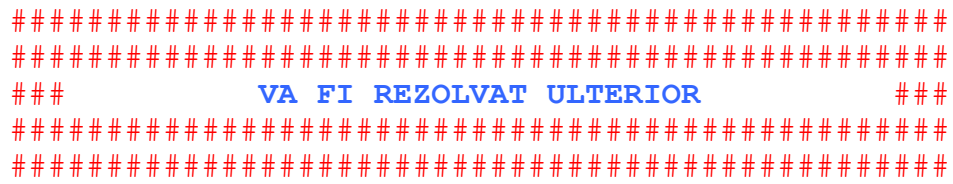
Varianta 82:

1. **2.**

3.



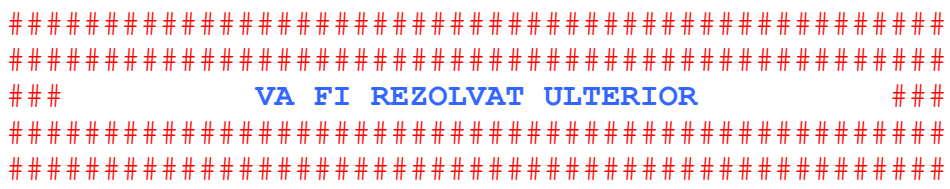
4.



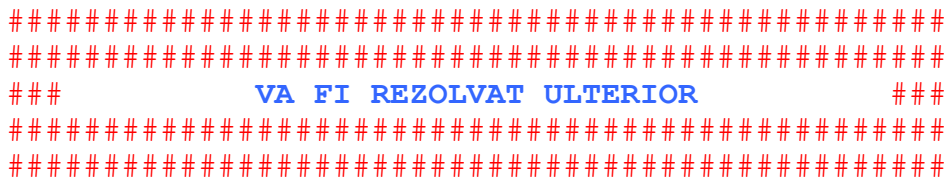
Varianta 83:

1. **2.**

3.



4.

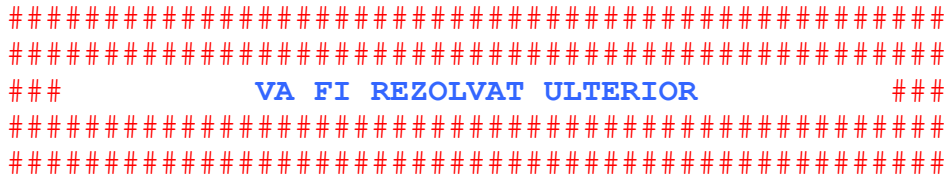


Varianta 84:

1. **2.**

3.

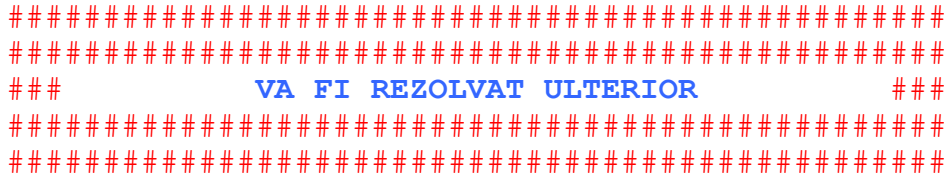
4.



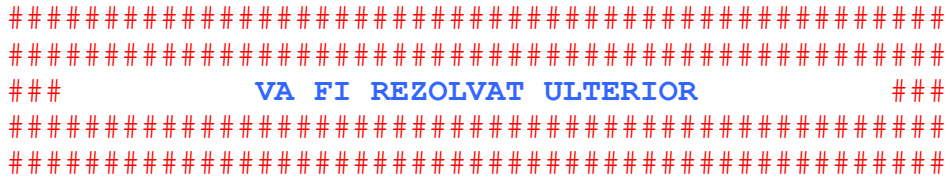
Varianta 85:

1. **2.**

3.



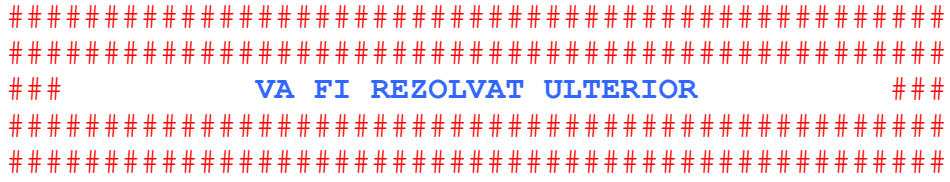
4.



Varianta 86:

1. **2.**

3.



4.

Varianta 87:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 88:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
```

```
##################################################
##################################################
```

Varianta 89:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

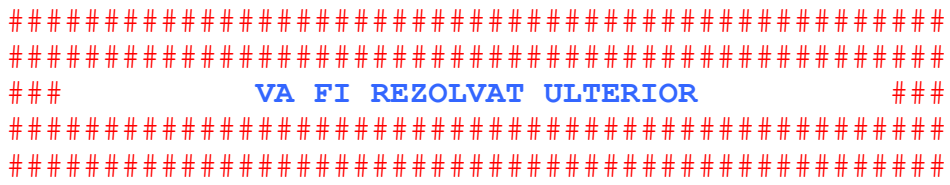
Varianta 90:

1. **2.**

3.

```
##################################################
##################################################
###           VA FI REZOLVAT ULTERIOR            ###
##################################################
##################################################
```

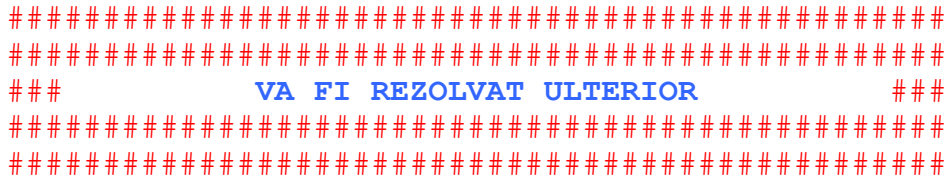
4.



Varianta 91:

1. **2.**

3.



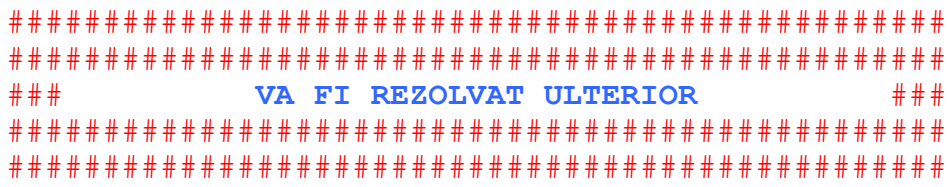
4.

Varianta 92:

1. **2.**

3.

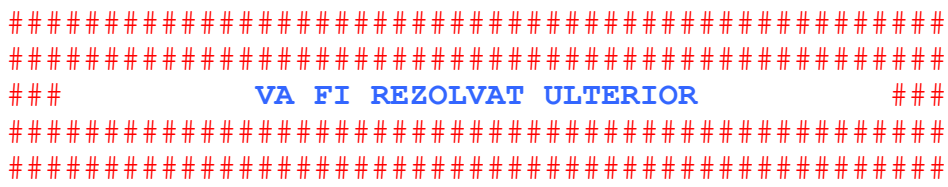
4.



Varianta 93:

1. **2.**

3.

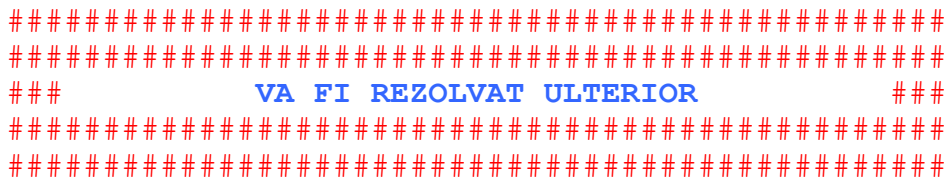


4.

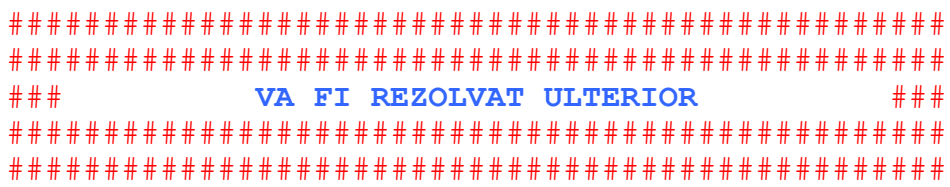
Varianta 94:

1. **2.**

3.



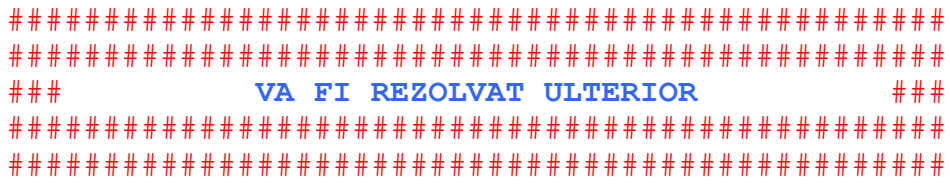
4.



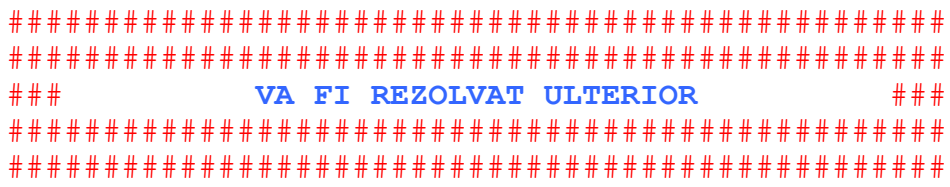
Varianta 95:

1. **2.**

3.



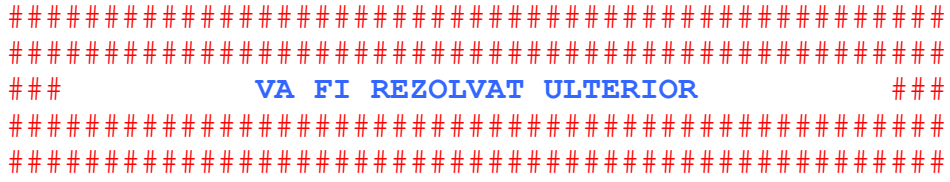
4.



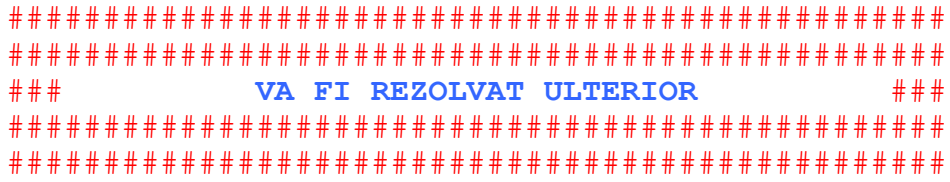
Varianta 96:

1. **2.**

3.



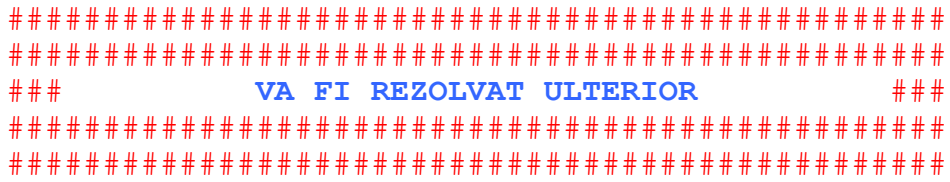
4.



Varianta 97:

1. **2.**

3.



4.

Varianta 98:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

Varianta 99:

1. **2.**

3.

```
##################################################
##################################################
###          VA FI REZOLVAT ULTERIOR           ###
##################################################
##################################################
```

4.

```
################################################
################################################
###          VA FI REZOLVAT ULTERIOR           ###
################################################
################################################
```

Varianta 100:

1. **2.**

3.

```
################################################
################################################
###          VA FI REZOLVAT ULTERIOR           ###
################################################
################################################
```

4.

```
################################################
################################################
###          VA FI REZOLVAT ULTERIOR           ###
################################################
################################################
```

```
################################################
################################################
###     RESTUL VOR FI REZOLVATE ULTERIOR !     ###
################################################
################################################
```

==========