PROIECT INDIVIDUAL LA INFORMATICĂ

TEMA: „METODA RELUĂRII” (BACKTRACKING)

REALIZAT: NICOARA CRISTIAN, clasa a XI-a „C”

VERIFICAT: MARIA GUȚU

IPLT „SPIRU HARET”

INFORMATIE:

Metoda reluării sau backtracking este o metodă de programare prin care se rezolvă problemele prin generarea fiecarei valori. În metoda reluării se presupune că soluţia problemei pe care trebuie să o rezolvăm poate fi reprezentată printr-un vector. Se pot folosi 3 tipuri ale acestei tehnici: se caută o soluție posibilă, se caută cea mai bună soluție, se caută toate soluțiile posibile.

AVANTAJE:

- ✓ Utilizează puțină memorie;
- ✓ Elimină atribuiri prin introducerea unor condiții;
- ✓ Folosește STIVA ca structura de memorie;
- ✓ Poate fi folostită la orice problema care necesită aflarea tuturor soluțiilor posibile;

DEZAVANTAJE:

- ✓ Necesită mult timp necesar pentru execuție;
- ✓ Majoritatea consideră această metodă dificilă

EXEMPLE DE PROGRAME:

1) **<u>Turnuri de cuburi</u>**

Se dau n cuburi numerotate 1,2,...,n, de laturi Li si culori Ci, i=1,2,...,n (fiecare culoare este codificata printr-un caracter). Sa se afiseze toate turnurile care se pot forma luând k cuburi din cele n disponibile, astfel încât:

-laturile cuburilor din turn sa fie in ordine crescatoare;

-culorile a oricare doua cuburi alaturate din turn sa fie diferite.

```
program cuburi;
type stiva=array [1..100] of integer;
var st:stiva;
i,n,p,k:integer;
a,ev:boolean;
L:array [1..10] of integer;
C:array [1..10] of char;
procedure init(k:integer;var st:stiva);
begin
st[k]:=0;
end;
procedure succesor(var a:boolean;var st:stiva;k:integer);
begin
if st[k]<n then
begin
st[k]:=st[k]+1;
a:=true;
end
else a:=false;
end;
procedure valid(var ev:boolean;st:stiva;k:integer);
var i:integer;
begin
ev:=true;
for i:=1 to k-1 do if L[st[k]]<=L[st[i]] then ev:=false;
if C[st[k]]=C[st[k-1]] then ev:=false;
end;
function solutie(k:integer):boolean;
begin
solutie:=(k=p);
end;
procedure tipar;
var i:integer;
begin
for i:=1 to p do write(st[i],' ');
writeln;
end;
begin
write('n= ');read(n);
write('p= ');read(p);
for i:=1 to n do
begin
write('L[',i,']=');readln(L[i]);
write('C[',i,']=');readln(C[i]);
end;
k:=1;init(k,st);
while k>0 do
begin
repeat
succesor(a,st,k);
if a then valid(ev,st,k);
until (not a) or (a and ev);
if a then if solutie(k) then tipar
else begin
k:=k+1;
init(k,st);
end
else k:=k-1;
end;
end.
```

2) Dintr-un număr de 6 cursuri opționale, un elev trebuie sa aleagă 3. Să se afișeze toate posibilitățile de alegere precum si numărul lor.

```
program cursuri;
const n=6;
p=3;
type stiva=array [1..10] of integer;
var st:stiva;
ev,a:boolean;
k:integer;
procedure init(k:integer;var st:stiva);
begin
if k>1 then st[k]:=st[k-1]
else if k=1 then st[k]:=0;
end;
procedure succesor(var a:boolean;var st:stiva;k:integer);
begin
if st[k]<n-p+k then begin st[k]:=st[k]+1;
a:=true;
end
else a:=false;
end;
procedure valid(var ev:boolean;var st:stiva;k:integer);
var i:integer;
begin
ev:=true;
for i:=1 to k-1 do if st[i]=st[k] then ev:=false;
end;
function solutie(k:integer):boolean;
begin
solutie:=(k=p);
end;
procedure tipar;
var i:integer;
begin
for i:=1 to p do write (st[i]);
writeln;
end;
begin;
k:=1;init(k,st);
while k>0 do
begin
repeat
succesor (a,st,k);
if a then valid(ev,st,k);
until (not a) or (a and ev);
if a then
if solutie(k) then tipar
else begin
k:=k+1;
init(k,st);
end
else k:=k-1;
end;
readln;
end.
```

3) **Numerele care îi plac lui Irinel**

Lui IRINEL îi plac nr. formate numai din cifre pare cifre aflate în ordine crescatoare. Sa se determine si sa se afiseze pe ecran toate numerele de n cifre (0<n<10) care îi plac lui Gigel. Valoarea lui n este un nr. natural care se citeste de la tastatura.

```
program nr_lui_IRINEL;
type stiva=array[1..100] of integer;
var st:stiva;
i,n,k:integer;
a,ev:boolean;
procedure init(k:integer;var st:stiva);
begin
st[k]:=-1;
end;
procedure succesor(var a:boolean;var st:stiva;k:integer);
begin
if st[k]<9 then begin st[k]:=st[k]+1;
a:=true;
end
else a:=false;
end;
procedure valid(var ev:boolean;st:stiva;k:integer);
var i:integer;
begin
ev:=true;
for i:=1 to k-1 do
if st[i] mod 2 <> 0 then ev:=false;
for i:=1 to k-1 do
if st[i]<st[i+1] then ev:=false;
end;
function solutie(k:integer):boolean;
begin
solutie:=(k=n);
end;
procedure tipar;
var i:integer;
begin
for i:=1 to n do write(st[i]);
writeln;
end;
begin
write('n= ');readln(n);
k:=1 ;init(k,st);
while k>0 do
begin
repeat
succesor(a,st,k);
if a then valid(ev,st,k);
until (not a) or (a and ev);
if a then if solutie(k) then tipar
else begin
k:=k+1;
init(k,st);
end
else k:=k-1;
```

```
end;
readln;
end.
```

4) PROGRAM PARTITI ALE UNUI NUMAR

```
program partitii_ale_unui_nr;
type stiva=array [1..10] of integer;
var st:stiva;
ev,a:boolean;
n,k:integer;
procedure init(k:integer;var st:stiva);
begin  st[k]:=0;
end;
procedure succesor(var a:boolean;var st:stiva;k:integer);
begin if st[k]<n then begin st[k]:=st[k]+1;
a:=true;
end
else a:=false;
end;
procedure valid(var ev:boolean;var st:stiva;k:integer);
var i,s:integer;
begin s:=0;
for i:=1 to k do
s:=s+st[i];
if s<=n then ev:=true
else ev:=false;
end;
function solutie(k:integer):boolean;
begin
solutie:=(k=n);
end;
procedure tipar;
var i:integer;
begin
for i:=1 to n do write (st[i]);
writeln;
end;
begin;
write ('n:=');readln (n);
k:=1;init(k,st);
while k>0 do
begin
repeat
succesor (a,st,k);
if a then valid(ev,st,k);
until (not a) or (a and ev);
if a then
if solutie(k) then tipar
else begin
k:=k+1;
init(k,st);
end
else k:=k-1;
end;
readln;
```

```
end.
```

5) Program Permutări

```
program permutari;
var st:array[1..25] of integer;i,n,p:integer;
procedure init;
var i:integer;
begin
write('N='); readln(n);for i:=1 to 25 do st[i]:=0;end;
function valid(p:integer):boolean;
var i:integer;
begin valid:=true;for i:=1 to p-1 do if st[i]=st[p] then valid:=false;end;
procedure tipar(p:integer);
var i:integer; begin for i:=1 to p do writeln(st[i],' ');end;
procedure back(p:integer);
 begin p:=1;
{plecam de la primul nivel }
st[p]:=0;
{initializam nivelul cu 0}
while p>0 do
{cat timp stiva nu este vida}
begin if st[p]<n then
{mai exista valori neincercate pe nivelul p}
begin st[p]:=st[p]+1;
{st[p]<-<o noua valoare din multimea valorilor posibile>}
if valid(p) then if p=n then tipar(p)
{solutia este finala}
else begin p:=p+1;
{trecem la nivelul urmator}
st[p]:=0;
{initializam valoarea de pe nivel cu 0}
end;end else
p:=p-1; {pas inapoi}
end;end;
begin
init;back(1);
end.
```

Concluzie:

Această tema este mai mult predestinată celora care cu siguranța vreau să parcurgă drumul IT, deoarece, după parerea

multor oameni, metoda reluării este destul de dificilă. Însă ea poate fi folositoarea în mai multe cazuri de probleme întrucât ea necesită puțină memorie. De asemenea ea antrenează logica atunci când alcătuim condițiile de verificare.

Bibliografie:

- ✓ MANUAL CLASA XI-a EDITURA Știința
- ✓ http://www.scritub.com/stiinta/informatica/METODA-BACKTRACKING1055131414.php
- ✓ https://mihaioltean.github.io/generarea_automata_a_programelor_de_calculator.pdf