

TEHNICA DE PROGRAMARE GREEDY

Proiect realizat de: Mihaela Frecăuțanu, clasa a XI-a C

INTRODUCERE

Metoda de programare **Greedy** se aplică problemelor de optimizare.

Aceasta metoda constă în faptul că se construieşte soluția optimă pas cu pas, la fiecare pas fiind selectat în soluție elementul care pare "cel mai bun/cel mai optim" la momentul respectiv, în speranța că această alegere locală va conduce la optimul global.

Un algoritm greedy este potrivit atunci când trebuie să luăm o serie de decizii, și anume pe cea mai conveabilă la un moment dat.



Se **iniţializează** mulţimea soluţiilor S egală cu mulţimea vidă, S=Ø

2

La fiecare pas se alege un anumit element **x∈A** (cel mai promiţător element la momentul respectiv) care poate conduce la o soluţie optimă

3

Se verifică dacă elementul ales poate fi adăugat la mulţimea soluţiilor:

- dacă da atunci va fi adăugat şi mulţimea soluţiilor devine S=S∪{x} - un element introdus în mulţimea S nu va mai putea fi eliminat
- Altfel el nu se mai testează ulterior

Procedeul continuă, până când au fost determinate toate elementele din mulţimea soluţiilor.

Această tehnică este folosită într-un mare număr de probleme și aplicații printre care:

Minimalizarea timpului mediu de așteptare Interclasarea optimă a șirurilor ordonate Coduri Huffman Cele mai scurte drumuri care pleacă din același punct Problema comis-voiajorului Arbori parțiali de cost minim

EXEMPLE de programe Pascal

Probleme rezolvate prin metoda Greedy

1) Problema continuă a rucsacului

Sarcina: Se consideră n objecte. Pentru fiecare object i (i=1, 2,...,n) se cunoaște greutatea gi și câștigul ci care se obține în urma transportului său la destinație. O persoană are un rucsac cu care pot fi transportate unul sau mai multe obiecte, greutatea sumară a cărora nu depășește valoarea Gmax. Elaborati un program care determină ce obiecte trebuie să transporte persoana în asa fel încât câstigul să fie maxim. În caz de necesitate, unele obiecte pot fi tăiate în fragmente mai mici.

Algoritm:



```
program rucsac;
                                                       begin
Var g:array [1..10] of integer;
                                                       aux:=q[i];
                                                       q[i]:=q[i+1];
 i,n,Gm,R, aux : integer;
                                                       σ[i+1]:=aux;
 ok:boolean;
                                                       ok:=false;
begin
                                                       end;
writeln('nr objecte'); readln(n);
                                                       end;
writeln('capacitate rucsac'); readln(R);
                                                      writeln; for i:=1 to n do write(g[i], '*');
writeln(' Obiectele de luat in rucsac:' );
                                                      Gm:=0:
for i:=1 to n do
                                                       i:=1;
 read (q[i]);
                                                       while (Gm + \sigma[i] \le R) do
ok:=false;
                                                       begin
while (ok=false) do
                                                       Gm:=Gm+g[i];
                                                       i := i+1;
begin
                                                       end:
ok:=true;
                                                       writeln('sunt', i-1, 'objecte cu greutate', Gm);
for i:=1 to n-1 do
                                                       writeln ( ' a ramas' , R-Gm,' loc liber' );
if q[i]>q[i+1] then
                                                       end.
```

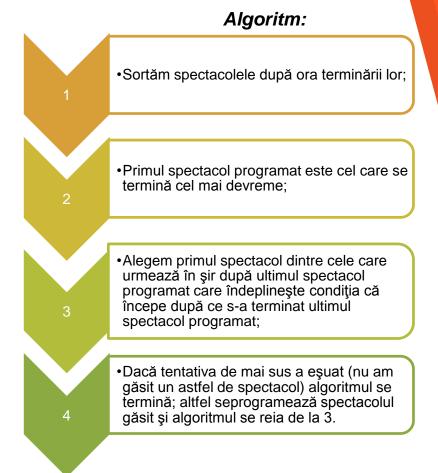
Răspuns:

```
OKHO BЫВОДА

nr obiecte
7
capacitate rucsac
15
Obiectele de luat in rucsac:
2
1
7
10
18
19
25
1*2*7*10*18*19*25*sunt 3 obiecte cu greutate 10
a ramas 5 loc liber
```

2) Problema spectacolelor

Sarcina: Într-un oraș de provincie se organizează un festival de teatru. Orașul are o singură sală de spectacole, iar la festival şi-au anunţat participarea mai multe trupe. Aşadar, în sală, într-o zi, trebuie planificate N spectacole. Pentru fiecare spectacol se cunoaște intervalul în care se desfășoară: [ora_inceput, ora_sfarsit]. Se cere să se planifice un număr maxim de spectacole care, bineînțeles, nu se pot suprapune.



```
Program spectacole;
                                                 readln(hh,mm);
Type spectacol=record
                                                 v[i].ora sf:=hh*60+mm;
 ora inc, ora sf:integer;
                                                 v[i].ord:=i;
ord:integer;
                                                 end;
 end;
                                                end;
Var v:arrav[1..30] of spectacol;
n, ultim, nr:integer;
                                                procedure greedy;
procedure sortare;
                                                var i:integer;
var i, j :integer; aux:spectacol;
                                                begin
begin
                                                 writeln('Ordinea spectacolelor este:');
 for i:=1 to n-1 do
                                                 ultim:=1;
 for j:=i+1 to n do
                                                 nr:=1;
 if v[j].ora sf < v[i].ora sf then</pre>
                                                 write(v[1].ord, ' ');
begin
                                                 for i:=2 to n do
 aux:=v[i];
v[i]:=v[i];
                                                 if v[i].ora inc>v[ultim].ora sf then
v[i]:=aux;
                                                 begin
 end;
                                                 write(v[i].ord,' ');
end;
                                                 ultim:=i;
procedure citire;
                                                 Inc(nr);
var hh, mm, i:integer;
                                                 end;
begin
                                                writeln('Se pot juca ', nr, ' spectacole');
 write('Numarul de spectacole:');
                                                end;
readln(n);
                                                begin
 for i:=1 to n do
 begin
                                                 citire;
 write('Spectacolul, i, incepe la:');
                                                 sortare;
 readln(hh,mm);
                                                greedy;
 v[i]. ora inc:=hh*60+mm;
                                                end.
write ('Spectacolul, i, se termina la:');
```

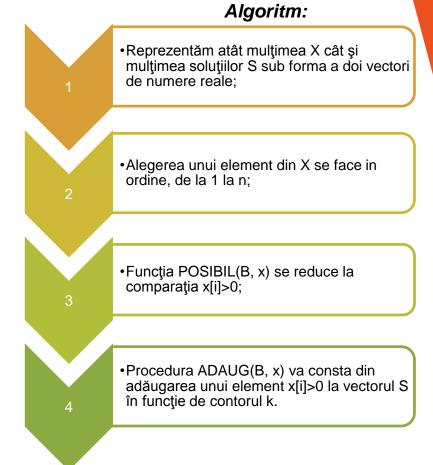
Răspuns:

Окно вывода

```
Numarul de spectacole:3
Spectacolul, i, incepe la:15
60
Spectacolul, i, se termina la:17
20
Spectacolul, i, incepe la:19
30
Spectacolul, i, se termina la:21
00
Spectacolul, i, se termina la:21
00
Spectacolul, i, incepe la:22
30
Spectacolul, i, se termina la:24
00
Ordinea spectacolelor este:
1 2 3 Se pot juca 3 spectacole
```

3) Problema sumei maxime

Sarcina: Se dă o mulţime X={x1, x2, . . ., xn } cu elemente reale. Să se determine o submulţime a lui X astfel încât suma elementelor submulţimii să fie maximă.



```
program suma maxima;
var s,x:array[1..20] of real;
i, k, n:integer;
begin
 write('Numarul de elemente n = ');
 readln(n);
 for i:=1 to n do
 begin
 write('x[',i,']= ');
 readln(x[i]);
 end;
 k := 0;
 for i:=1 to n do
 if x[i]>0 then
 begin
 k := k+1;
 s[k]:=x[i]
 end;
 for i:=1 to k do
 write(s[i]:5:2,' ');
 readln;
end.
```

Răspuns:

Окно вывода

```
Numarul de elemente n = 5

x[1]= 2

x[2]= 6

x[3]= 7

x[4]= 8

x[5]= 9

2.00 6.00 7.00 8.00 9.00
```

5) Problema cu submulțimea

Sarcina: Elementele sale sunt numere reale, iar cel puţin unul din ele satisface condiţia ai>0. Elaboraţi un program care determină o submulţime B, astfel încât suma elementelor din B să fie maximă.

```
Program Problema;
{ Tehnica Greedy
const nmax=1000;
var A : arrav [1..nmax] of real;
n : 1..nmax;
B : array [1..nmax] of real;
m : 0..nmax;
x : real;
i : 1..nmax;
Function ExistaElemente : boolean;
var i : integer;
begin
ExistaElemente:=false;
for i:=1 to n do
if A[i]>0 then ExistaElemente:=true;
end; { ExistaElemente }
procedure AlegeUnElement(var x : real);
var i : integer;
begin
i:=1:
while A[i] <= 0 do i:=i+1;
x := A[i];
A[i]:=0;
end; { AlegeUnElement }
procedure IncludeElementul(x : real);
begin
```

```
m := m+1;
B[m]:=x;
end; { IncludeElementul }
begin
write('Dati n='); readln(n);
writeln('Daţi elementele mulţimii A:');
for i:=1 to n do read(A[i]);
writeln:
m := 0 :
while ExistaElemente do
begin
AlegeUnElement(x);
IncludeElementul(x);
end;
writeln('Elementele multimii B:');
for i:=1 to m do writeln(B[i]);
readln:
end.
```



Окно вывода

```
Daţi n=6
Daţi elementele mulţimii A:
1
2
91
283
32
375
Elementele mulţimii B:
1
2
91
283|
32
375
```



Concluzie:

Metoda Greedy nu caută să determine toate soluțiile posibile (care ar putea fi prea numeroase) și apoi să aleagă din ele pe cea optimă, ci caută să introducă direct un element x în soluția optimă. Acest lucru duce la eficiența algorimilor Greedy, însă nu conduce în mod necesar la o soluție optimă și nici nu este posibilă formularea unui criteriu general conform căruia să putem stabili exact dacă metoda Greedy rezolvă sau nu o anumită problemă de optimizare. Acest motiv duce la însoțirea fiecărei rezolvări prin metoda Greedy a unei demonstrații matematice(în general prin inducție).



BIBLIOGRAFIE:

- http://www.worldit.info/articole/algoritmica-articole/metoda-greedy
- https://sites.google.com/site/eildegez/home/clasa-xi/prezentarea-metodeigreedy
- https://www.scribd.com/doc/43454385/Metoda-Greedy#download
- https://www.infoarena.ro/metoda-greedy-si-problema-fractionara-a-rucsacului
- Anatol Gremalschi, Informatică Manual pentru clasa a 11-a, Știința, 2014