

# **PROIECT INDIVIDUAL LA INFORMATICĂ TEMA: METODA GREEDY**

A REALIZAT: Proca Virgiliu

A VERIFICAT: Maria Guțu

**Metoda Greedy** este una dintre cele mai directe tehnici de proiectare a algoritmilor care poate fi aplicată la o gamă largă de probleme. În general, această metodă se aplică problemelor de optimizare. Majoritatea acestor probleme constau în determinarea unei submulțimi  $B$ , a unei mulțimi  $A$  cu  $n$  elemente care să îndeplinească anumite condiții pentru a fi acceptată. Orice astfel de submulțime care respectă aceste restricții se numește soluție posibilă. Din mulțimea tuturor soluțiilor posibile se dorește determinarea unei soluții care maximizează sau minimizează o funcție de cost. O soluție posibilă care realizează acest lucru se numește soluție optimă. Considerăm că soluțiile posibile au următoarea proprietate: dacă  $B$  este o soluție posibilă, atunci orice submulțime a sa este soluție posibilă. Specificul acestei metode constă în faptul că se construiește soluția optimă pas cu pas, la fiecare pas fiind selectat (sau "înghițit") în soluție elementul care pare "cel mai bun" la momentul respectiv, în speranța că va duce la soluția optimă globală.

### *Pentru obținerea soluției Greedy*

- Se sortează elementele din  $S$  în măsura descreșterii corespunderii criteriului  $C$ . Se obține șirul sortat:  $s^*1, s^*2, \dots, s^*N$ .
- Se consideră soluția inițial vidă  $B$
- Se adaugă consecutiv în  $B$  elementele  $s^*1, s^*2, \dots, s^*i, \dots$  atât timp cât nu se încalcă restricțiile  $R$  ale problemei  $P$ .

*Schema generală a unui algoritm bazat pe metoda Greedy poate fi redată cu ajutorul unui ciclu:*

```
while ExistăElemente do  
begin  
    AlegeUnElement( $x$ );  
    IncludeElementul( $x$ );  
end;
```

- Problema P1;

Se consideră mulțimea  $A=\{a_1, a_2, \dots, a_i, \dots, a_n\}$  elementele căreia sînt numere reale, iar cel puțin unul din ele satisface condiția  $a_i > 0$ . Elaborați un program care determină o submulțime  $B, B \subseteq A$ , astfel încît suma elementelor din  $B$  să fi e maximă.

```
Program P1;
{ Tehnica Greedy }
const nmax=1000;
var A : array [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('Dați n='); readln(n);
    writeln('Dați elementele mulțimii A:');
    for i:=1 to n do read(A[i]);
    writeln;
    m:=0;
    while ExistaElemente do
        begin
            AlegeUnElement(x);
            IncludeElementul(x);
        end;
    writeln('Elementele mulțimii B:');
    for i:=1 to m do writeln(B[i]);
    readln;
end.
```

- Problema 2;

Scrieți un program, care afișează modalitatea de plată, folosind un număr minim de bancnote, a unei sume întregi S de lei ( $S < 20000$ ). Plata se efectuează folosind bancnote cu valoarea 1, 5, 10, 50, 100, 200 și 500 de lei. Numărul de bancnote de fiecare valoare se citește din fișierul text BANI.IN, care conține 7 rânduri, în fiecare din care sunt indicate numărul de bancnote respectiv de 1, 5, 10, 50, 100, 200 și 500 de lei.

```
Program bani;
type tablou=array[1..3,1..7] of integer;
var s,ss,i : integer; a:tablou; f:text;
{In primul rind al tabelului vom pastra nominalul bancnotelor}
{In al doilea rind - numarul bancnotelor citite din fisier}
{In al treilea rind - numarul bancnotelor obtinute la schimb}
Procedure Afisare(i:integer;sa:integer);
begin writeln('suma ',s);
if sa<>0
then writeln('nu poate fi transformata cu bancnotele date ')
else
begin writeln('se plateste cu urmatoarele bancnote');
for i:=1 to 7 do
if a[3,i]<>0
then writeln('bancnote de ',a[1,i]:6,' sau folosit ',a[3,i]);
end
end; { Afisare }
Procedure calcul(var sa:integer);
var nb:integer;
begin
i:=7;
while (i>=1) and (sa>0) do
begin nb:=sa div a[1,i];
if nb<>0 then if nb>= a[2,i]
then a[3,i]:=a[2,i]
else a[3,i]:=nb;
sa:=sa-a[3,i]*a[1,i];
i:=i-1;
end;
end; { calcul }
begin
a[1,1]:=1; a[1,2]:=5; a[1,3]:=10; a[1,4]:=50;
a[1,5]:=100; a[1,6]:=200; a[1,7]:=500;
assign (f,'bani.in'); reset(f);
for i:=1 to 7 do readln(f,a[2,i]);
write ('introduceti suma de lei S ');readln(s);
ss:=s; calcul(ss);
Afisare(ss);
end.
```

- Problema 3;

Datele se citesc din fișierul text data.in cu următoarea structură: prima linie a fișierului conține un număr întreg N – numărul de segmente. Următoarele N linii conțin câte 2 numere întregi, separate prin spațiu – abscisa extremității stângi  $X_i$  a segmentului i și lungimea lui  $L_i$ .

```

type segment=record st,dr : integer; end;
sets = array[1..100] of segment;
var a,b: sets;
n,k: integer;
procedure readdata(var x: sets; var n: integer);
var f: text;
i,r: integer;
begin
  assign(f, 'data.in'); reset(f);
  readln(f,n);
  for i:=1 to n do
  begin
    readln(f, x[i].st, r);
    x[i].dr:=x[i].st+r;
  end;
  close(f);
end;
procedure sort (var x:sets; n:integer);
var i,j: integer;
t: segment;
begin
  for i:=1 to n -1 do
  for j:=1 to n-i do
  if x[j].dr>x[j+1].dr then
begin t:=x[j]; x[j]:=x[j+1]; x[j+1]:=t; end;
end;
procedure solve(var y:sets; x:sets; n:integer;
var k:integer);
var i: integer;
begin
y[1]:=x[1]; k:=1;
for i:=2 to n do
  if x[i].st > y[k].dr then begin k:=k+1; y[k]:=x[i]; end;
end;
procedure print(x: sets;k:integer);
var i: integer;
begin
  writeln(k);
  for i:=1 to k do writeln(x[i].st, ' ',x[i].dr);
end;
begin
readdata(a,n);
sort(a,n);
solve(b,a,n,k);
print(b,k);
end.

```

- Problema 4;

Într-o școală de șoferie se organizează lecții teoretice. Școala are doar un cabinet, dar are mai multe grupuri de studenți. Așadar într-un cabinet într-o zi trebuie planificate N lecții. Pentru fiecare lecție se cunoaște ora de început și sfârșit. Sa se planifice un număr maxim de lecții care nu se suprapun.

```
Program spectacole;
Type spectacol=record
    ora_inc, ora_sf:integer;
    ord:integer;
end;
Var v:array[1..30] of lectie;
n, ultim, nr:integer;
procedure sortare;
var i,j :integer; aux:lectie;
begin
    for i:=1 to n-1 do
        for j:=i+1 to n do
            if v[j].ora_sf < v[i].ora_sf then
                begin
                    aux:=v[j];
                    v[j]:=v[i];
                    v[i]:=aux;
                end;
            end;
        end;
    end;
end;
procedure citire;
var hh, mm, i:integer;
begin
    write('Numarul de lectii:');
    readln(n);
    for i:=1 to n do
        begin
            write('Lectia, i, incepe la:');
            readln(hh,mm);
            v[i].ora_inc:=hh*60+mm;
            write('Lectia, i, se termina la:');
            readln(hh,mm);
            v[i].ora_sf:=hh*60+mm;
            v[i].ord:=i;
        end;
    end;
end;
procedure greedy;
var ;integer;
begin
    writeln('Ordinea lectiilor este:');
    ultim:=1;
    nr:=1;
    write(v[1].ord,' ');
    for i:=2 to n do
        if v[i].ora_inc>v[ultim].ora_sf then
            begin
                write(v[i].ord,' ');
                ultim:=i;
                Inc(nr);
            end;
        end;
    writeln('Se pot organiza ', nr, ' lectii');end;
begin
    citire; sortare; greedy;
end.
```

- Problema 5: Un șofer are un camion în care poate transporta o greutate maximă **G** și urmează să efectueze un transport în urma căruia va primi un câștig. Șoferul are la dispoziție **n** obiecte și cunoaște pentru fiecare obiect greutatea și câștigul care se obține în urma transportului la destinație. Se cere să se precizeze ce obiecte trebuie să transporte șoferul în așa fel încât câștigul să fie maxim și care este câștigul

```

program camion_greedy;
type coloana=array[1..5] of real;
    camion=array[1..100] of coloana;
var r:camion; n:integer; g:real;

procedure citire_obiecte (var a:camion; var nr_ob:integer; var gr:real);
var f:text;
begin
    assign(f,'camion.in');    reset(f);
    readln(f,gr);  {citește greutatea totală}
    nr_ob:=0;
    while not eof(f) do
        begin
            Inc(nr_ob);
            readln(f,a[nr_ob,1],a[nr_ob,2],a[nr_ob,3]    );
            a[nr_ob,4]:=a[nr_ob,3]/a[nr_ob,2]; {se calculează eficiența}
        end;
    close(f);
end;

procedure afisare(var x:camion; n:integer);
var i,j:integer;
begin
    writeln('obiect   ','greutate ','castig   ','eficienta');
    for j:=1 to n do
        begin
            for i:=1 to 4 do
                write(x[j,i]:6:2,' ');
            writeln;
        end;
end;

procedure sortare(var x:camion; n:integer);
var i,j:integer; c:coloana;
begin
    for i:=1 to n-1 do
        for j:=i+1 to n do
            if x[i,4]<x[j,4] then
                begin
                    c:=x[i];
                    x[i]:=x[j];
                    x[j]:=c;
                end;
end;

procedure greedy(var x:camion; n:integer; gr_max:real);
var castig_total, gr_camion, rest:real;
    nr_ob:integer;
begin
    castig_total:=0;    gr_camion:=0;    nr_ob:=1;
    while (gr_camion<gr_max) and (nr_ob<=n) do
        begin
            if x[nr_ob,2]<=gr_max-gr_camion then

```

```

var castig_total, gr_camion, rest:real;
    nr_ob:integer;
begin
    castig_total:=0;    gr_camion:=0;    nr_ob:=1;
    while (gr_camion<gr_max) and (nr_ob<=n) do
        begin
            if x[nr_ob,2]<=gr_max-gr_camion then
                begin
                    gr_camion:=gr_camion+x[nr_ob,2];
                    castig_total:=castig_total+x[nr_ob,3];
                    x[nr_ob,5]:=100;
                end
            else
                begin
                    rest:=gr_max-gr_camion;
                    castig_total:=castig_total+rest*x[nr_ob,4];
                    x[nr_ob,3]:=rest*x[nr_ob,3]/x[nr_ob,2];
                    gr_camion:=gr_camion+rest;
                    x[nr_ob,5]:=rest*100/x[nr_ob,2];
                    x[nr_ob,2]:=rest;
                end;
                Inc(nr_ob);
            end;
        if gr_camion<gr_max then
            writeln('Camionul nu este plin')
        else
            begin
                writeln('In camion au fost incarcate obiectele:');
                writeln('cod obiect  greutate  castig  eficienta  procent');
                for nr_ob:=1 to n do
                    if x[nr_ob,5]<>0 then
                        writeln(x[nr_ob,1]:9:2,x[nr_ob,2]:10:2,x[nr_ob,3]:8:2,x[nr_ob,4]:11:2,x[nr_ob,5]:10:2);
                        writeln('Castigul total=',castig_total:10:2);
                    end;
                end;
            end;
        end;
    end;
begin
    citire_obiecte(r,n,g);
    writeln('Inainte de sortare');
    afisare(r,n);
    sortare(r,n);
    writeln('Dupa sortare');
    afisare(r,n);
    greedy(r,n,g);
    readln;
end.

```



## ***Concluzie***

- Algoritmii Greedy sunt caracterizati de metoda lor de functionare: la fiecare pas se alege cel mai bun candidat posibil, dupa evaluarea tuturor acestora. Metoda determina intotdeauna o singura solutie, asigurand un optim local, dar nu intotdeauna si global. Tehnica Greedy este una de optimizare, ruland mai rapid decat un Backtracking, dar nefiind intotdeauna cea mai buna.
- Cand nu aveti o idee mai buna legata de o problema, in timpul unui concurs, o implementare Greedy ar putea aduce in jur de 30% din punctaj. Exista situatii in care algoritmii clacheaza, cum ar fi problema comisului voiajor, sau problemele NP-complete.
- Metoda Greedy are si avantaje: poate fi aplicata multor probleme: determinarea celor mai scurte drumuri in grafuri (Dijkstra), determinarea arborelui minimal de acoperire (Prim, Kruskal), codificare arborilor Huffmann, planificarea activitatilor, problema spectacolelor si problema fractionara a rucsacului. Dintre acestea, articolul le trateaza numai pe ultimele doua pentru a da un exemplu cat mai bun a modului de functionare si aplicare a algoritmilor Greedy.