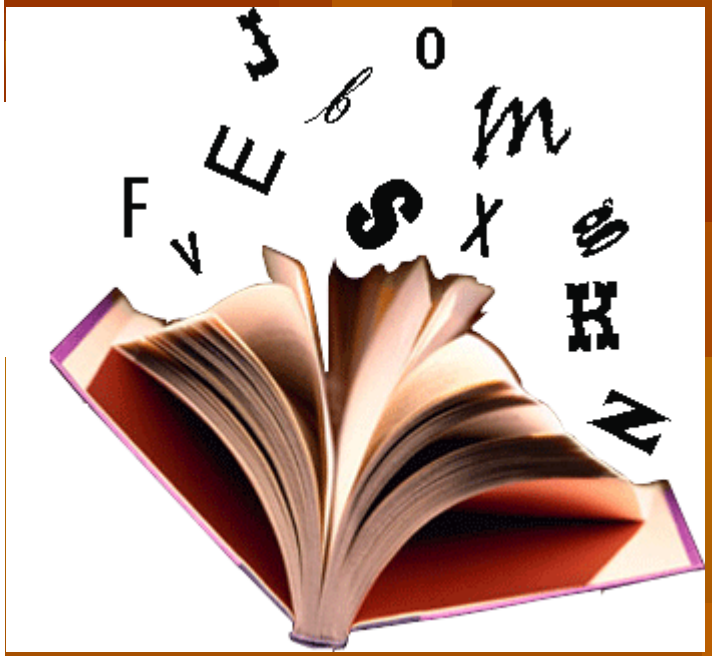
PROIECT LA INFORMATICA



Clasa a XI-a I1;

3 March 2009

PROGRAMAREA DINAMICA

Programarea dinamica face parte dintr-o categorie de metode matematice numite metode de scufundare.O problemã de programare dinamicã contine subprobleme comune (care nu sunt independente) din a cãror rezolvare si combinare se obtine solutia unei (sub)probleme mai mari.Deci rezolvarea unei (sub)probleme se poate face doar dupã solutionarea (sub)problemelor, care alcãtuiesc problema respectivã.Spunem cã programarea dinamica actioneaza de jos in sus,prelucrând si combinând subcazuri mici,obtinând astfel solutia pentru subcazuri tot mai mari.

- Totodatã programarea dinamicã evitã rezolvarea de mai multor ori a aceleiasi subprobleme(rezolvându-se doar problemele independente),prin memorarea rezultatelor partiale.
- Nu există un criteriu pe baza căruia să identificăm cu siguranţă o problemă pentru rezolvarea căreia trebuie să utilizăm metoda programării dinamice, dar putem formula două proprietăţi care sugerează o soluţie prin programare dinamică.

PRINCIPII FUNDAMENTALE ALE PROGRAMARII DINAMICE

- Programarea dinamica, ca si metoda divide et impera, rezolva problemele combinand solutiile subproblemelor. Dupa cum am vazut, algoritmii divide et impera partitioneaza problemele in subprobleme independente, rezolva subproblemele in mod recursiv, iar apoi combina solutiile lor pentru a rezolva problema initiala. Daca subproblemele contin subsubprobleme comune, in locul metodei divide et impera este mai avantajos de aplicat tehnica programarii dinamice.

- Sa analizam insa pentru inceput ce se intampla cu un algoritm divide et impera in aceasta din urma situatie. Descompunerea recursiva a cazurilor in subcazuri ale aceleiasi probleme, care sunt apoi rezolvate in mod independent, poate duce uneori la calcularea de mai multe ori a aceluiasi subcaz, si deci, la o eficienta scazuta a algoritmului.

- Primul principiu de baza al programarii dinamice: evitarea calcularii de mai multe ori a aceluiasi subcaz, prin memorarea rezultatelor intermediare.
- Al doilea principiu fundamental al programarii dinamice este faptul ca ea opereaza de jos in sus (bottom-up). Se porneste de obicei de la cele mai mici subcazuri. Combinand solutiile lor, se obtin solutii pentru subcazuri din ce in ce mai mari, pina se ajunge, in final, la solutia cazului initial.

- al treilea principiu fundamental, programarea dinamica este utilizata pentru a optimiza o problema care satisface principiul optimalitatii: intr-o secventa optima de decizii sau alegeri, fiecare subsecventa trebuie sa fie de asemenea optima. Cu toate ca pare evident, acest principiu nu este intotdeauna valabil si aceasta se intampla atunci cand subsecventele nu sunt independente, adica atunci cand optimizarea unei secvente intra in conflict cu optimizarea celorlalte subsecvente.

Agricultorul

- **Pentru a mari productia de porumb un agricultor trebuie sa elimine toti porumbii care se afla la distanta mai mica decat un X dat fata de vecinii sai. Determinati porumbii care trebuie sa ramana cunoscand locul in care au iesit printr-o singura coordonata pe axa OX data.**

DESCRIERE

- Folosim programare dinamica;
- Construim vectorul l, l[i] reprezinta numarul maxim de porumbi pastrand porumbul I si eliminare incepe de la i+1;
- Productia maxima este egala cu l[1];
- Plecand de la primul porumb gasim primul de dupa el la distanta X care are productia de la el egala cu max-1;

```
type vector=array [-1..100] of integer;
var a,l:vector;p,n,h,i,j,max:integer;
procedure citire;
var i:integer;
begin
readln(n,h);
for i:=1 to n do
read(a[i]);
end;
begin
citire;
l[n]:=1;
for i:=n-1 downto 1 do begin
max:=0;
for j:=i+1 to n do
if (a[j]-a[i]>=h) and (l[j]>max) then max:=l[j];
l[i]:=1+max;end;
writeln(l[1]);
p:=1; max:=l[1];
while max>0 do
begin
write(a[p],' ');
dec(max);
for i:=p+1 to n do
if (a[i]-a[p]>=h) and (l[i]=max) then break;p:=i;
end;
end.
```

Sursa executabila

```
type vector=array[1..10000] of integer;
var a, v:vector;
 var i ,j,max,n,gmax,x,p:integer;
 f,g:text;
 begin

 assign(f,'porumbi.in'); reset(f);
 assign(g,'porumbi.out');rewrite(g);
  read(f,n,x);
   for i:= 1 to n do
    read(f,v[i]);
    a[n]:=1;
     for i := n-1 downto 1 do

      begin
      max:=0;
       for j:= i+1 to n do
        if (v[j]-v[i]>=x)and(a[j]>max) then
         max:=a[j];
          a[i]:=max+1;
           end;
            writeln(g,v[1],' ');
             p:= 1;
              max:=a[1]-1;

              for i:= 2 to n do
               if (v[i]-v[p]>=x) and(a[i]=max) then
                begin
                 write(g,v[i],' ');
                  p:=i;
                   dec(max);
                    end;
                  close(g);
                 end.
```

Sursa executabila

X=2

N=5

2 5 6 9 10

Solutii

X=2

N=5

A:2 5 6 9 10

L:3 2 2 1 1

BIBLIOGRAFIE

- CULEGERE DE PROBLEME PENTRU CLASA AXI-A;
- CAIET DE INFORMATICA;
- INTERNET;

SFARSIT.