IPLT “Spiru Haret “

Lucrare practică

TEMA: Metoda trierii

Efectua de : Grubleac
Gabriela ,eleva clasei a 11-a

Verificat de : Guțu Maria ,
profesor de informatică

Descrierea aspectelor teoretice

Se numeşte metoda trierii o metodă ce indentifică toate soluţiile unei probleme în dependenţă de mulţimea soluţiilor posibile.

Schema generală:

```
for i:= 1 to k do   if SolutiePosibila(si)
 then PrelucrareaSolutiei(si)
```

unde `si` este solute analizata. `SolutiePosibila` este o funcţie booleană care returnează valoarea true dacă elementul `si` satisface condiţiile problemei şi false în caz contrar. `PrelucrareaSolutiei` este o procedură care efectuează prelucrarea elementului selectat.

Probleme din cotidian care pot fi rezolvate utilizând această metodă:

- aflarea numărului minim de monede care pot fi date drept plată sau rest;
- medicii deseori se confruntă cu necesitatea aplicării metodei trierii cazurilor, când numărul răniților sau bolnavilor este foarte mare, medicul fiind suprasolicitat, în cazul unui război, sau când își periclitează propria viață în cazul unei epidemii periculoase;
- aflarea ariei maxime a unui lot de teren, avînd la dispoziție o anumită lungime de sîrmă ghimpată, spre exemplu (ca perimetru dat);
- generarea submulțimilor unei mulțimi (aflarea tuturor combinațiilor posibile), ceea ce ne poate fi foarte util în viața de zi cu zi;
- afișarea coordonatelor a două puncte date ce au distanță minimă sau maximă, ceea ce va fi foarte folositor dacă plănuim o călătorie;
- calcularea șanselor de a lua premiul mare la loterie etc.

Pentru a aplica metoda trierii, în alte țări se folosește cel mai des tipul de algoritm *Greedy,* care are rolul de a construi soluția optimă pas cu pas, la fiecare pas fiind selectat în soluție elementul care pare optim la momentul respectiv.

Avantajele și dezavantajele metodei

Avantajul principal al algoritmilor bazaţi pe metoda trierii constă în faptul că programele respective sînt relativ simple, iar depanarea lor nu necesită teste sofisticate.

Un dezavantaj ar fi faptul că algoritmii bazaţi pe metoda trierii se calculează, implicit sau explicit, mulţimea soluţiilor posibile S. În problemele relativ simple elementele din mulţimea soluţiilor posibile pot fi enumerate direct. În problemele mai complicate generarea soluţiilor posibile necesită elaborarea unor algoritmi speciali , cea ce denotă că această metodă nu este eficientă pentru programele complexe . De asemenea sarcina programatorului fiind alcătuirea unui algoritm special ,un algoritm exponențial care nu poate prelucra o cantitate mare de date.

Utilizarea metodei în rezolvarea problemelor

1. Scrieți un program care primind ca parametru un număr natural n, returnează valoarea true dacă n este prim și false în caz contrar.

```
  var  n : integer;

function SolutiePosibila(n:integer):boolean;
begin
if (n mod 2 =0 ) or  (n mod 3 =0 ) or  (n mod 5 =0 ) or (n<>2)or (n<>3)or
(n<>5) then SolutiePosibila:=false
else SolutiePosibila:=true ;{conditia aproximativa si simplificata a numerelor prime}
end;
procedure PrelucrareaSolutiei1;
begin
  Writeln('true');
end;
procedure PrelucrareaSolutiei2;
begin
  Writeln('false');
end;
begin
write('Daţi n='); readln(n);
      if SolutiePosibila(n) then PrelucrareaSolutiei1
         else  PrelucrareaSolutiei2;
      end.
```

2. Se consideră numerele naturale din mulţimea {0, 1, 2, ..., n}. Elaboraţi un program care determină pentru cîte numere K din această mulţime produsul caror cifre este egală cu m.

```
type Natural=0..MaxInt;
var i, K, m, n : Natural;
function ProdusulCifrelor(i:Natural):Natural;
var produsul : Natural;
begin  produsul:=1;
repeat    produsul:=produsul*(i mod 10);{operatia de inmultire a cifrelor }
i:=i div 10;{se va repeta pana cand I va fi 0}
until i=0;
produsulCifrelor:=produsul;
end;
```

```
function SolutiePosibila(i:Natural):boolean;
begin
if produsulCifrelor(i)=m then SolutiePosibila:=true
else SolutiePosibila:=false;
end;
procedure PrelucrareaSolutiei(i:Natural);
begin
writeln('i=', i);
K:=K+1;
end; { PrelucrareaSolutiei }
begin
write('Daţi n='); readln(n);
write('Daţi m='); readln(m);
K:=0;
for i:=0 to n do
      if SolutiePosibila(i) then PrelucrareaSolutiei(i);
      writeln('K=', K);  readln;

                                          end.
```

3. Se consideră mulţimea P = {P1, P2, …, Pn} formată din n puncte (2<=n <= 30) pe un plan euclidian. Fiecare punct Pj este defi nit prin coordonatele sale xj, yj. Elaboraţi un program care afi şează la ecran coordonatele punctelor Pa, Pb distanţa dintre care este maximă.

```
const nmax=30;
 type Punct = record
         x, y : real;
 end;
 Indice = 1..nmax;
var P : array[Indice] of Punct;
 j, m, n : Indice;
 dmax : real; { distanţa maxima }
 PA, PB : Punct;
function Distanta(A, B : Punct) : real;
 begin
 Distanta:=sqrt(sqr(A.x-B.x)+sqr(A.y-B.y));{se calculeaza distanta dupa formula datata}
 end;
 function SolutiePosibila(j,m:Indice):boolean;
 begin
 if j<>m then SolutiePosibila:=true
 else SolutiePosibila:=false;
 end;
 procedure PrelucrareaSolutiei(A, B : Punct);
 begin  if Distanta(A, B)>dmax then
    begin
    PA:=A; PB:=B;{comparea distantelor}
    dmax:=Distanta(A, B);
    end;
    end;
    begin
    write('Dati n=');
    readln(n);
```

```
    writeln('Daţi coordonatele x, y ale punctelor');
    for j:=1 to n do
      begin
      write('P[', j, ']: '); readln(P[j].x, P[j].y);{ciclul petru introducerea
coordonatelor punctelor }
      end;
      dmax:=0;
      for j:=1 to n do     { creerea unei matrici bidimensionale petru determinarea distantei maxime }
        for m:=1 to n do
            if SolutiePosibila(j, m) then
                        PrelucrareaSolutiei(P[j], P[m]);
                        writeln('Soluţia: PA=(', PA.x:5:2, ',', PA.y:5:2,
')');
                        writeln('         PB=(', PB.x:5:2, ',', PB.y:5:2,
')');
                        readln; end.
```

4. Scrieți un program care enumeră tote numerele pozitive între două valori date .

```
var i, n ,k,m:integer;
 function SolutiePosibila(i:integer):boolean;
 begin  if (i)>0 then SolutiePosibila:=true
 else SolutiePosibila:=false;{conditia petru numerele pozitive}
 end;
 procedure PrelucrareaSolutiei(i:integer);
 begin
 writeln('i=', i);{afisarea numarului}
 K:=K+1;{incrementarea variabilei ce numara valorile}
 end;
 begin
 write('Daţi n='); readln(n);
 write('Daţi m='); readln(m);
 K:=0;
 for i:=n to m do
       if SolutiePosibila(i) then PrelucrareaSolutiei(i);
       writeln('K=', K);  readln;{creerea unui ciclu ce va testa si afisa valorile cerute}
             end.
```

5. Elaborați un program ce enumeră toate numerele pozitive între două valori date .

```
var i, n ,k,m:integer;
 function SolutiePosibila(i:integer):boolean;
 begin  if i mod 2=0 then SolutiePosibila:=true
 else SolutiePosibila:=false; {conditia petru numerele pare}
 end;
 procedure PrelucrareaSolutiei(i:integer);
 begin
 writeln('i=', i); {afisarea numarului}
 K:=K+1; {incrementarea variabilei ce numara valorile}
 end;
 begin
 write('Daţi n='); readln(n);
 write('Daţi m='); readln(m);
```

```
 K:=0;
 for i:=n to m do
       if SolutiePosibila(i) then PrelucrareaSolutiei(i);
       writeln('K=', K);   ;{creerea unui ciclu ce va testa si afisa valorile cerute}
end.
```

Concluzii

Metoda trierii este convenabilă de a fi folosită petru programe simple , ce nu necesită algoritmi speciali și o mare varietate de soluții posibile. De aceea metoda este folosită doar în scopuri didactice, în realitate fiind rar întrebuințată.

Bibliografie

- Manualul de informatică clasa a 11-a
- https://www.slideshare.net/foegirl/metoda-trierii-33371122
- timofti7.simplesite.com/435052344

Concluzie