Lucrare individualăla informatică

Tehnica de programare “desparte și stăpânește”

Autor: Cazacu Felicia

Evaluator: Maria Guțu

# Metoda de programare DIVIDE ET IMPERA constĂ În ÎmpărțiRea

# problemei iniȚiale de dimensiuni [n] În douĂ sau mai multe probleme de dimensiuni reduse, În general se executĂ ÎmpărȚirea În douĂ subprobleme de dimensiuni aproximativ egale Și anume [n/2] . Împărțirea în subprobleme are loc pÂnă cÂnd dimensiunea acestora devine suficient de micĂ pentru a fi rezolvate În mod direct(cazul de bazĂ), Dupa rezolvarea celor douĂ subprobleme se executĂ faza de combinare a rezultatelor În vederea rezoLvării Întregii probleme .

# DupĂ cum sugereaza Și numele "desparte si stĂpânește "etapele rezolVării unei probleme (numita problema iniȚiAlă) În DIVIDE ET IMPERA sunt :

# -descompunerea problemei iniȚiale În subprobleme independente ,smilare problemei de bazĂ ,de dimensiuni mai mici

# -descompunerea treptatĂ a subproblemelor În alte subprobleme din ce În ce mai simple, Până cÂnd se pot rezolva imediat, prin algoritmul simplificat ;

# -rezolvarea subproblemelor simple ;

# -combinarea soluȚiilor gĂsite pentru construirea soluȚiilor subproblemelor de dimensiuni din ce În ce mai mari

# -combinarea ultimelor soluȚii determinĂ obȚinerea soluȚiei problemei iniȚiale.

**Avantaje:**

* Presupune un timp de execuție mai mic
* Această metodă stă la baza altor metode de sortare rapidă

**Dezavantaje**

* Nu este o metodă universală, astfel nu toate problemele permit divizarea
* Nu este o metodă simplă, pentru majoritate.

**Exemple probleme:**

daca i=j, valoarea maxima va fi v[i];

> contrar vom imparti vectorul in doi vectori: primul vector va contine componentele de la i la (i+j) div 2, al doilea vector va contine componentele de la (I+J) div 2 +1 la j; rezolvam problemele (aflam maximul pentru fiecare din ele) iar solutia problemei va fi data de valoarea maxima dintre rezultatele celor doua subprobleme.

program maxim;

var v:array[1..10] of integer;

n,i:integer;

function max(i,j:integer):integer;

var a,b:integer;

begin

if i=j then max:=v[i]

else begin

a:=max(i, (i+j) div 2);

b:=max((i+j) div 2+1,j);

if a>b then max:=a

else max:=b;

end;

end;

begin

write(’n=’);

readln(n);

for i:=1 to n do read(v[i]);

writeln(maximul este ’,max(1,n));

end.

Fie n valori numere naturale a1,a2,a3,..,an. Determinati cel mai mare divizor comun al lor prin metoda Divide Et Impera. Se imparte sirul a1,a2,a3,..,an in doua subsiruri a1,a2,a3,..,am,respectiv am+1,am+2,.,an,unde m reprezinta pozitia de mijloc,m=(1+n) div 2.

Cmmdc(a1,a2,..,an)= Cmmdc(a1,a2,..,am), Cmmdc(am+1,am+2,.,an))

program cmmdc\_sir;

const nmax=20;

type indice=1..nmax;

var a:array[indice] of word;

n:indice;

procedure citire;

var i:indice;

begin

readln(n);

for i:=1 to n do read(a[i]);

end;

function euclid(x,y:word):word;

var r:word;

begin

while y<>0 do

begin

r:=x mod y;

x:=y;

y:=r;

end;

euclid:=x;

end;

function cmmdc(p,q:indice):word;

var m:indice;

begin

if q-p<=1 then cmmdc:=euclid(a[p],a[q])

else

begin

m:=(p+q) div 2;

cmmdc:=euclid(cmmdc(p,m),cmmdc(m+1,q));

end;

end;

begin

citire;

writeln(’cmmdc=’,cmmdc(1,n));

readln;

end.

Se considera un sir cu n elemente, initial toate egale cu n. Se imparte sirul pe jumatate, elementele din jumatatea stanga incrementandu-se, iar cele din jumatatea dreapta decrementandu-se cu o unitate. Daca exista element 'nepereche' exact la mijloc acesta ramane neschimbat. Celor doua jumatati li se aplica acelasi 'tratament' si jumatatilor jumatatilor la fel etc. pana cand se obtin secvente de cate un element.

program codare;

var a:array[1..100] of integer;

n,i:integer;

procedure codare(p,q):integer;

var m,i:integer;

begin

if q-p=2 then begin a[q]:=a[q]-1;

a[p]:=a[p]+1;

end

else if q-p=1 then begin a[q]:=a[q]-1;

a[p]:=a[p]+1;

end

else if (q-p) mod 2=0 then

begin m:=(p+q) div 2;

for i:=p to m-1do a[i]:=a[i]+1;

for i:=m+1 to q do a[i]:=a[i]-1;

codare(p,m-1);

codare(m+1,q);

end

else

begin m:=(p+q) div 2;

for i:=p to m do a[i]:=a[i]+1;

for i:=m+1 to q do a[i]:=a[i]-1;

codare(p,m);

codare (m+1,q);

end;

end;

begin

readln(n);

for i:=1 to n do a[i]:=n;

codare(1,n);

for i:=1 to n do write(a[i]:4);

writeln;

end.

Sa se verifice daca o valoare data x exista intr-un sir de numere intregi ordonate crescator. Se va folosi un algoritm de cautare bazat pe metoda Divide Et Impera.

Se descompune problema in doua subprobleme de acelasi tip. Se imparte vectorul in doi subvectori: primul subvector va contine elementele pana la pozitia de mijloc, iar al doilea va fi alcatuit din elementele de dupa pozitia din mijloc. Verificam daca valoarea cautata se gaseste chiar pe pozitia din mijloc si in caz afirmativ cautarea se opreste. Daca valoarea cautata este mai mica decat elementul situat pe pozitia din mijloc, cautarea trebuie continuata in subvectorul stang, iar daca este mai mare, cautarea continua in subvectorul drept.

program cautare;

type vector=array[1..20] of integer;

var v:vector;n,x,i:integer;

function caut(p,q:integer):integer;

var mij:integer;

begin

if q<p then caut:=-1

else

begin

mij:=(p+q) div 2;

if v[mij]=x then caut:=mij

else if x<v[mij] then caut:=(p,mij-1)

else caut:=caut(mij+1,q);

end;

end;

begin

write(’n=’);

readln(n);

write(’v[1]=’);

readln(v[1]);

for i:=2 to n do

repeat

write(’v[’,i,’]=’);

readln(v[i]);

until v[i]>v[i-1];

write(’x=’);

readln(x);

writeln(caut(1,n));

end.

**Bibliografie:**

<http://www.creeaza.com/referate/informatica/Metoda-de-programare-DIVIDE-ET449.php>

<http://prohorenco.blogspot.com/2017/05/tehnici-de-programare-desparte-si.html?m=1>

<http://www.scritub.com/stiinta/informatica/METODA-DIVIDE-ET-IMPERA25186243.php>