PROIECT INDIVIDUAL LA INFORMATICĂ NR.2

TEMA: „Metoda *desparte* şi *stăpâneşte*”

(în latină *divide et impera*)

REALIZAT: NICOARA CRISTIAN, clasa a XI-a „C”

VERIFICAT: MARIA GUȚU

INFORMATIE:

Metoda de programare desparte si stăpânește constă în împărțirea problemei inițiale de dimensiuni [n] în două sau mai multe probleme de dimensiuni reduse .În general se execută împărțirea in două subprobleme de dimensiuni aproximativ egale si 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).După rezolvarea celor două subprobleme se execută faza de combinare a rezultatelor în vederea rezolvării întregii probleme .

AVANTAJE:

- ✓ Timpul execuției relativ mic ;
- ✓ Programele sunt ușor de înțeles ;
- ✓ Această metodă stă la baza a mai multe metode de sortarea rapidă ;

- ✓ Această tehnică admite o implementare recursive ;

DEZAVANTAJE:

- ✓ Această metodă poate fi aplicată numai atunci când prelucrarea cerută admite divizarea problemei curente în subprobleme de dimensiuni mai mici ;
- ✓ Tehnica este folosită destul de rar întrucât ea necesită mai multe condiții ;
- ✓ Mulți socot această metodă complicată ;

ETAPELE METODEI DESPARTE ȘI STĂPÂNEȘTE:

- ✓ 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

SCHEMA GENERALĂ:

```
procedure DesparteSiStapineste(i, j : integer; var x : tip);
var m : integer;
    x1, x2 : tip;
begin
   if SolutieDirecta(i, j) then Prelucrare(i, j, x)
      else
         begin
           m:=(j-i) div 2;
           DesparteSiStapineste(i, i+m, x1);
           DesparteSiStapineste(i+m+1, j, x2);
           Combina(x1, x2, x);
         end;
end;
```

EXEMPLE DE PROGRAME:

1) Maxim intr-un vector

Se citeste un vector cu n componente, numere naturale. Se cere sa se tipareasca valoarea maxima.
program cuburi;

```
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.
```

2) Cel mai mare divizor comun

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.

```
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.
```

3) Codare Sir

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 codare1;
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.
```

4) Partitionarea unui sir

Se considera un sir de n numere intregi, ordonat crescator si un numar intreg x. Sa se partitioneze sirul dat in doua subssiruri, astfel incat toate elementele primului sir sa fie mai mici decat x,

iar toate elementele celui de-al doilea sir sa fie mai mari decat x. Se va folosi un algoritm Divide Et Impera.

```
 program
partitionare;
type vector=array[1..20] of integer;
var v:vector;n,x,i,ref:integer;
function part(p,q:integer):integer;
var mij:integer; begin
if q<p then part:=p else begin
mij:=(p+q) div 2; if x=v[mij] then
part:=mij else if x<v[mij] then
part:=part(p,mij-1) else
part:=part(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); ref:=part(1,n);
writeln('primul vector'); for i:=1
to ref-1 do write (v[i]:5); writeln;
writeln('al doilea vector'); for
i:=ref to n do write(v[i]:5);
readln; end.
```

5) Pozitia k

Se citeste de la tastatura un sir de n elemente numere intregi.Sa se gaseasca elementul aflat pe o pozitie data k in sirul ordonat crescator, fara a efectua ordonarea.

```
program pozitia_k; const
nmax=20; type indice=1..nmax;
var a:array[indice] of integer;
n,k:indice; procedure citire;
var i:indice; begin
write('n='); readln(n);
for i:=1 to n do begin write('a[',i,']=');
readln(a[i]); end; end;
procedure afisare;
var i:indice; begin
```

```
writeln('vectorul este'); for
i:=1 to n do write(a[i]:4);
writeln; end;
function divide(p,q:indice):indice;
var st,dr:indice;x:integer; begin
st:=p; dr:=q; x:=a[p]; while st<dr
do begin
while (st<dr) and (a[dr]>=x) do dec(dr);a[st]:=a[dr];
while (st<dr) and (a[dr]<=x) do inc(st);a[dr]:=a[st];
end; a[st]:=x; divide:=st; end;
function qselect(st,dr,k:indice):integer;
var p:indice; begin p:=divide(st,dr);
if k=p-st+1 then qselect:=qselect(st,p-1,k)
else qselect:=qselect(p+1,dr,k-(p-st+1));
end; begin citire; write('k='); readln(k);
writeln('pozitia ',k,' in vectorul sortat este ',qselect(1,n,k));
afisare; end.
```

Concluzie:

Consider că această metodă este una destul de importantă pentru persoanele care urmează o cale în programare sau în IT, întrucât ea este folositoare în multe cazuri, precum sortarea rapidă prin metode ca quicksort sau bubblesort. Multe persoane susțin că această metodă este destul de dificila, ea fiind opțională în curriculum.

Bibliografie:

- ✓ MANUAL CLASA XI-a EDITURA Știința
- ✓ http://www.creeaza.com/referate/informatica/Metoda-de-programareDIVIDE-ET449.php
- ✓ http://prohorenco.blogspot.com/2017/05/tehnici-de-programaredesparte-si.html
- ✓ http://www.scritub.com/stiinta/informatica/METODA-DIVIDE-ETIMPERA25186243.php