TEOREME de programare

Clara Ionescu

După ce am făcut cunoștință cu "teoremele" numite *Prelucrarea unui șir de date*, *Decizia*, *Selecția* și *Căutarea (secvențială)*, urmează să învățăm cum se implementează subalgoritmii de *numărare* (contorizare).

Numărarea

Fie următoarele probleme propuse spre rezolvare:

P12. Cunoscând numărul familiilor dintr-un județ, valoarea veniturilor lor, precum și valoarea "coșului minim", să se stabilească numărul familiilor care trăiesc sub venitul minim de trai!

P13. Se consideră rezultatele *Olimpiadei Naționale de Informatică* și punctajul minim necesar pentru a putea participa la barajul pentru a intra în lotul național. Să se calculeze procentul participanților care vor intra la baraj!

P14. Să se numere câte vocale conține un text dat!

Datele de intrare pentru problemele de mai sus sunt șiruri de valori (numere sau caractere). Rezultatul cerut este un număr care reprezintă câte elemente din șir au o anumită proprietate. Enunțurile nu garantează că există cu siguranță un astfel de element, deci soluția furnizată poate fi și 0.

Având în vedere că fiecare element trebuie verificat, deoarece este posibil ca (și) ultimul să aibă proprietatea cerută, vom lucra cu o structură repetitivă de tipul **Pentru**.

Vom prezenta în continuare câteva detalii privind declarațiile din programul care va conține subalgoritmul care rezolvă o asemenea problemă:

```
n: întreg          {numărul elementelor șirului de prelucrat}
x: tablou(1..n): tip element   {elementele șirului dat}
bucăți: întreg                 {rezultatul}
Subalgoritm Numărare(n,x,buc):
  buc := 0
  Pentru i=1,n execută
    Dacă P(X[i]) atunci
      buc := buc + 1
  Sfârșit pentru
Sfârșit subalgoritm
```

În cele ce urmează vom prezenta implementarea subprogramelor care rezolvă problemele **P12-P14**.

```
{avem definit tipul Tablou = array[1..10000] of Real}
{și tipul Membri = array[1..10000] of Byte, pentru un}
{vector cu numărul membrilor celor n familii}
procedure P12(n:Byte; X:Tablou; nr:Membri;
                      min:Real; var buc:Byte);
var i:Byte;           {min = venitul minim de trai}
begin
  buc:=0;
  for i:=1 to n do
    if X[i]/nr[i] < min then
      Inc(buc)
end;
```

P13. Această problemă cere calcularea unui procent, dar acesta se realizează, bineînțeles, tot pe baza unei numărări. De data aceasta, vom implementa subalgoritmul `Numărare` cu ajutorul unei funcții.

```
function procent(n:Byte; X:Tablou; pct:Byte):
                                          Real;
{avem definit tipul Tablou = array[1..250] of Byte}
var buc, i:Byte;
begin
  buc:=0;{încă nu am găsit nici un elem. cu propr. cerută}
  for i:=1 to n do {pct = punctaj necesar pentru baraj}
    if X[i] >= pct then
      Inc(buc); {numărul elem. având propr. cerută}
  procent:=(100*buc)/n {calcularea procentului cerut}
end;
```

P14.

```
function nrvoc(text:string; vocale:mult):Byte;
{avem definit tipul mult = set of Char, iar variabila vocale conține mulțimea vocalelor, litere mici și mari}
var buc,i:Byte;
begin
  buc:=0;
  for i:=1 to Length(text) do
    if text[i] in vocale then
      Inc(buc);
  nrvoc:=buc
end;
```