dragoș maria balan
george balan

culegere de probleme pentru concursuri

LIMBAJ BASIC

1992

Coperta :
C. arh. Brînduşa Luchian

Referent ştiinţific :
şef lucrări Şt. Gh. Pentiuc
Universitatea „Ştefan cel Mare" — Suceava

DRAGA - MARIA BALAN
GEORGE BALAN

LIMBAJUL BASIC

CULEGERE DE PROBLEME PENTRU CONCURSURI

Referent științific : Șef de lucrări Gh. Șt. Pentiuc
Universitatea "ȘTEFAN CEL MARE"
SUCEAVA

DRAGA - MARIA BALAN
GEORGE BALAN

LIMBAJUL BASIC

CULEGERE DE PROBLEME PENTRU CONCURSURI

1

EDITURA LICURICI
SUCEAVA
1992

CUVÎNT ÎNAINTE

Culegerea de față se vrea a fi prima dintr-un ciclu de culegeri pentru învățarea eficientă a limbajului BASIC devenit foarte popular și datorită răspîndirii în România a calculatoarelor din familia HC și a microcalculatoarelor compatibile IBM-PC.

Ca orice limbaj de programare, și acesta, necesită pentru o însușire temeinică un lucru intens pe calculator, alcătuirea și rularea unui număr cît mai mare de programe.

Culegerea pune la dispoziție modele și tehnici de alcătuire a programelor, noțiunile matematice utilizate nedepășind, în general nivelul gimnazial de pregătire. Deși acest lucru ar da impresia că problemele ar fi simple, în realitate problemele propuse și rezolvate demonstrează contrariul.

Stilul de exprimare este riguros și precis, dar comentariile la rezolvările problemelor sînt pe înțelesul tuturor.

Cu un calculator în față și cu puțină imaginație, aceste probleme deschid un larg cîmp de lucru pentru alte probleme.

Lucrarea conține două părți :

- prima parte conține 40 probleme propuse care acoperă șirurile de numere, șirurile de caractere, operații cu numere naturale, probleme diverse.

- a doua parte conține soluțiile problemelor comentate integral, în cazul problemelor mai dificile sau doar programul în cazul problemelor mai ușoare.

Seria de culegeri care debutează cu cea de față vrea să acopere tematic un cîmp cît mai vast de probleme.

Culegerea se adresează elevilor de gimnaziu, celor de liceu, studenților din primii ani de studiu și tuturor iubitorilor de informatică.

SUCEAVA, iulie 1992 *Autorii*

PARTEA I

PROBLEME PROPUSE

1. PROBLEME CU ŞIRURI DE NUMERE

1.1. Se consideră două mulţimi de numere A şi B, A cu n elemente şi B cu m elemente, unde m şi n sînt numere naturale mai mari decît 1. Să se facă un program BASIC prin care să se determine mulţimile : $A \cup B$, $A \cap B$, $A - B$, $A \times B$, $A \Delta B$.

1.2. Se consideră un şir[1] A de numere cu n elemente distincte două cîte două. Să se facă un program BASIC prin care să se ordoneze acest şir în ordine crescătoare (descrescătoare) şi să se determine locul fiecărui element în şirul iniţial şi în şirul astfel ordonat.

1.3. Acelaşi text ca la problema 1.2., însă şirul poate avea elemente care se pot repeta. În acest caz să se indice un şir în care elementele apar o singură dată şi să se indice pentru fiecare element al acestui şir toate poziţiile în care apare acesta în şirul iniţial şi în şirul ordonat crescător, respectiv descrescător.

Să se rezolve problema poziţiilor elementului în şirurile ordonate şi fără a face ordonarea şirurilor.

1.4. Se consideră un şir A de numere cu n elemente. Să se facă un program BASIC prin care să se introducă un număr de la tastatură şi să se determine dacă acesta se găseşte în şirul dat şi, în caz ....

1) Pe tot parcursul lucrării vom înţelege prin şir o mulţime finită, ordonată. În multe lucrări de informatică această noţiune este denumită eronat prin *vector*, ceea ce nu corespunde ideii de spaţiu vectorial.

afirmativ, să se indice de cîte ori apare elementul respectiv şi toate poziţiile acestuia în următoarele şiruri : cel iniţial, cel ordonat crescător şi cel ordonat descrescător, fără a face efectiv ordonarea.

1.5. Se consideră un şir A de numere cu n elemente. Să se determine toate anagramele acestui şir. (Prin anagramă înţelegem un nou şir care are aceleaşi elemente cu şirul iniţial şi fiecare element apare exact de atîtea ori ca şi în şirul iniţial).

1.6. Se consideră o mulţime A de numere cu n elemente. Să se construiască un şir care să conţină pe primele k poziţii ($k < n$) elemente ale mulţimii A ordonate crescător, iar pe următoarele n-k poziţii celelalte elemente ordonate descrescător. Se vor indica toate posibilităţile.

1.7. Se consideră o mulţime A de numere cu n elemente şi $k<n$ un număr natural. Să se determine toate submulţimile distincte de numere cu k elemente din A.

1.8. Fie A un şir de numere cu n elemente şi $k<n$ un număr natural. Să se construiască un nou şir care să conţină pe primele k locuri ultimele k elemente ale şirului iniţial, iar pe ultimele n-k locuri primele n-k elemente ale şirului iniţial.

1.9. Fie A un şir de n numere.

a) Să se calculeze suma termenilor acestui şir ;

b) să se calculeze produsul termenilor acestui şir ;

c) să se determine dacă termenii şirului sînt în progresie aritmetică ;

d) să se determine dacă termenii acestui şir sînt în progresie geometrică ;

e) să se calculeze media armonică a elementelor acestui şir;

La punctele c) şi d) se va specifica raţia în caz că răspunsul este afirmativ.

1.10. Se consideră un şir A de numere cu n elemente şi un număr

natural $k < n$. Să se găsească elementul de pe poziția k în șirul ordonat crescător (descrescător), fără a ordona șirul.

1.11. Se consideră un șir A de numere cu n elemente. Să se determine elementul maxim (minim) din șir și toate pozițiile în care acesta apare printr-o singură parcurgere completă a șirului.

1.12. Se consideră un șir de numere A cu n elemente. Să se determine :

a) cîte elemente sînt pozitive, negative și cîte sînt nule ;

b) cîte sînt întregi și cîte nu.

2. PROBLEME CU ȘIRURI DE CARACTERE.

2.1. Se consideră un șir de caractere introdus de la tastatură. Să se determine lungimea sa și codurile ASCII ale caracterelor șirului.

2.2. Se consideră un șir de caractere și k un număr natural. Să se cerceteze dacă k este mai mic decît lungimea șirului și, în caz afirmativ, să se afișeze cele k elemente din mijloc.

2.3. Cum putem introduce caracterele <CR> și <"> de la tastatură în cadrul unui șir de caractere ?

2.4. Să se determine numărul de litere mari, litere mici, cifre și caractere speciale dintr-un șir de caractere.

2.5. Să se determine dacă un caracter introdus de la tastatură se găsește într-un șir și, apoi, să se determine toate pozițiile în care se găsește acesta în șirul considerat.

2.6. Să se determine dacă un șir de caractere este subșir al altui șir de caractere și să se precizeze poziția de început.

2.7. Prin *cuvînt* înțelegem o succesiune de litere, din care numai prima literă poate fi majusculă și care nu conține spații sau caractere speciale. Să se extragă toate cuvintele dintr-un șir de caractere dat.

2.8. Să se anagrameze un cuvînt care conține numai litere mari.

2.9. Să se transforme toate literele mici ale unui text în litere mari.

2.10. Se dă un număr natural de maximum opt cifre. Să se construiască un șir de caractere care să exprime în litere numărul considerat.

2.11. Se consideră un număr natural de maxim patru cifre. Să se construiască un șir de caractere care să reprezinte numărul considerat în cifre romane.

2.12. Se consideră un șir de caractere care reprezintă un număr scris cu cifre romane. Să se exprime numărul în scrierea arabă.

3. PROBLEME CU NUMERE NATURALE.[1)]

3.1. Se consideră un număr natural *n*. Să se facă un program BASIC pentru scrierea lui într-o bază de numerație *b* dată.

3.2. Să se facă un program BASIC care să efectueze cele patru operații cu fracții. Rezultatul va fi scris sub formă de fracție ireductibilă. Operațiile vor fi efectuate utilizînd numai algoritmii învățați în clasa a V-a.

3.3. Să se facă un program pentru calculul sumei numerelor reprezentate de cifrele unui număr dat. Nu este permisă introducerea sau utilizarea șirurilor de caractere pentru a defini numere.

3.4. Utilizînd numai criteriile de divizibilitate din clasa a V-a, să se stabilească dacă un număr natural *n* se divide prin 2, 3, 4, 5, 9, 25.

1) În cadrul acestui paragraf, numerele naturale care intervin vor fi presupuse strict mai mari decît 1.

3.5. Se cunoaşte următorul criteriu de divizibilitate prin 11 a unui număr natural : *Un număr natural n se divide prin 11 ⇔ suma numerelor reprezentate de cifrele de pe locurile pare, din care se scade suma numerelor reprezentate de cifrele de pe locurile impare, este un număr care se divide prin 11.* Să se facă un program care să utilizeze acest criteriu pentru a stabili dacă un număr natural dat se divide sau nu la 11.

3.6. Să se facă un program cu ajutorul căruia să se efectueze cele patru operaţii cu numere mari (oricîte cifre). În cazul împărţirii se va specifica cîtul şi restul.

3.7. Să se calculeze 1000 ! (prin n ! înţelegem produsul numerelor naturale de la 1 la n, unde n este un număr natural).

3.8. Să se descompună un număr natural n dat în factori primi.

3.9. Să se determine toate numerele prime mai mici decît un număr natural dat.

3.10. Să se determine toate numerele naturale perfecte mai mici decît un număr natural dat (prin număr perfect înţelegem un număr natural în care suma tuturor divizorilor pozitivi ai numărului este egală cu dublul numărului considerat).

3.11. Să se determine toţi divizorii unui număr natural dat.

3.12. Să se găsească toate soluţiile în numere naturale, mai mici decît 200 ale ecuaţiei :

$$5 \cdot x - 3 \cdot y = 1.$$

4. PROBLEME DIVERSE.

4.1. Să se facă un program prin care să se efectueze cele patru

operații cu numere naturale scrise în baza de numerație 7, fără a trece numerele în baza 10. La împărțire se va indica cîtul și restul.

4.2. Să se determine toate numerele naturale de forma $\overline{a_1a_2...a_n}$ scrise în baza 10 care sînt egale cu $a_1!+a_2!+...+a_n!$.

4.3. Să se rezolve următorul rebus aritmetic:

ARAD + SATU + MARE + ARGES = JUDETE

4.4. Să se determine toate numerele naturale prime cu un număr natural n, mai mici decît n.

PARTEA A II - A

REZOLVAREA PROBLEMELOR

1. PROBLEME CU ŞIRURI DE NUMERE.

1.1. Se consideră două mulţimi de numere A şi B, A cu n elemente şi B cu m elemente, unde m şi n sînt numere naturale mai mari decît 1. Să se facă un program BASIC prin care să se determine mulţimile : $A \cup B$, $A \cap B$, $A - B$, $A \times B$, $A \Delta B$..

Problema în sine nu este dificilă. Programul cuprinde două etape :

- iniţializarea, introducerea şi validarea datelor;
- efectuarea reuniunii, intersecţiei, diferenţei, produsului cartezian şi a diferenţei simetrice a celor două mulţimi.

În prima etapă, după ştergerea ecranului şi iniţializarea zonei variabilelor (10), se introduce numărul elementelor mulţimii A (20), după care se verifică dacă A astfel construită este mulţime, adică dacă elementele nu se repetă. Recomandăm ca aceste validări să se facă întotdeauna cînd se introduc date. Aceste operaţii sînt realizate în liniile 30 - 80. În mod analog, liniile 90 - 150, sînt introduse şi elementele mulţimii B.

Reuniunea este efectuată de instrucţiunile de la liniile 210 - 310. Pentru aceasta, într-o mulţime C sînt trecute mai întîi elementele mulţmii A. Sînt trecute apoi în mulţimea C toate elementele din B care nu se găsesc în A.

Pentru *intersecţie* (liniile 320 - 390) se afişează toate elementele din A care se găsesc şi în B, iar pentru *diferenţă* (liniile 400 - 520) se păstrează toate elementele lui A care nu se găsesc în B.

Produsul cartezian este realizat imediat prin două cicluri prin instrucţiunile de la liniile 530 - 550.

Diferența simetrică este realizată simplu observînd că aceasta este tocmai diferența dintre reuniunea și intersecția celor două mulțimi (liniile 560 - 670).

Programul, pentru calculatoarele compatibile IBM - PC, este următorul :

Se șterge ecranul și se inițializează zona de date :

```
10 CLS : CLEAR
```

Se introduce numărul de elemente ale mulțimii A și elementele acestei mulțimi :

```
20 INPUT "Introduceti numarul de elemente ale multimii
A :"; N : DIM A(N)
30 FOR I=1 TO N
```

Se validează elementele mulțimii A (să nu existe elemente care să se repete). Se observă instrucțiunea NEXT J, I care închide două cicluri :primul după J, iar al doilea după I. Aceasta este valabil doar în cazul calculatoarelor care folosesc limbajul GWBASIC, pentru calculatoarele compatibile SPECTRUM, cum ar fi cele de producție românească din seria HC, trebuie dată cîte o instrucțiune NEXT pentru închiderea fiecărui ciclu.

```
40 PRINT"A("; I ; ")="; : INPUT A(I)
50 FOR J=1 TO I - 1
60 IF A(I)<>A(J) THEN GOTO 80
70 PRINT "elementul se repeta" : GOTO 40
80 NEXT J, I
```

Se introduce numărul de elemente ale mulțimii B și elementele acestei mulțimi :

```
90 INPUT   "Introduceti numarul de elemente ale multimii
B :"; M : DIM B(M)
100 FOR I=1 TO M
110 PRINT"B("; I ; ")="; : INPUT B(I)
```

Se validează elementele mulțimii B (să nu existe elemente care să se repete) :

```
120 FOR J=1 TO I - 1
130 IF B(I)<>B(J) THEN GOTO 150
140 PRINT "elementul se repeta" : GOTO 110
150 NEXT J,I
```

Se stabilește numărul maxim de elemente ale reuniunii celor două mulțimi, precum și două tablouri unidimensionale care să conțină elementele mulțimilor rezultate în urma efectuării operațiilor cerute.

```
160 L=N+M
170 DIM D(L), C(L)
```

Se șterge ecranul și se afișează cele două mulțimi inițiale :

```
190 CLS : PRINT "A = {"; : FOR I=1 TO N : PRINT A(I); :
NEXT I : PRINT"}"
200 PRINT "B = {"; : FOR I=1 TO M : PRINT B(I); : NEXT I
: PRINT"}"
```

Se pregătește efectuarea reuniunii celor două mulțimi :

```
210 PRINT "REUNIUNEA : " : PRINT : PRINT "C = {";
220 L=0
```

Se păstrează în C elementele mulțimii A :

```
230 FOR I=1 TO N : L=L+1 : C(L)=A(I) : NEXT I
```

Se introduc în C elementele mulțimii B care nu se găsesc în A :

```
240 FOR J=1 TO M : K=0
250 FOR I=1 TO N
260 IF B(J)=A(I) THEN K=1
270 NEXT I
280 IF K<>0 THEN GOTO 300
290 L=L+1 : C(L)=B(J)
300 NEXT J
```

Se afișează elementele reuniunii :

```
310 FOR I=1 TO L : PRINT C(I); : NEXT I : PRINT"}" : PRINT
```

Se pregătește efectuarea intersecției celor două mulțimi :

```
320 PRINT "INTERSECTIA : " : PRINT"D = {";
```

Se afișează elementele intersecției, care sînt toate elementele din A ce nu se găsesc și în B :

```
330 FOR I=1 TO N : K=0
340 FOR J=1 TO M
350 IF A(I)=B(J) THEN K=1
360 NEXT J
370 IF K=1 THEN PRINT A(I);
380 NEXT I
390 PRINT "}" : PRINT
```

Se pregătește efectuarea diferenței celor două mulțimi :

```
400 PRINT "DIFERENTA : " : PRINT "E = {";
```

Se determină elementele diferenței, care sînt toate elementele din A ce nu se găsesc în B :

```
410 L=0
420 FOR I=1 TO N : K=0
430 FOR J=1 TO M
440 IF A(I)=B(J) THEN K=1
450 NEXT J
460 IF K<>0 THEN GOTO 480
470 L=L+1 : D(L)=A(I)
480 NEXT I
```

Se afișează elementele diferenței :

```
490 FOR I=1 TO L
500 PRINT D(I);
510 NEXT I
520 PRINT "}" : PRINT
```

Se afișează elementele produsului cartezian :

```
530 PRINT "PRODUSUL CARTEZIAN : " : PRINT "F = {";
540 FOR I=1 TO N : FOR J=1 TO M : PRINT"("; A(I); B(J);
")"; : NEXT J, I
550 PRINT"}" : PRINT
```

Se pregătește efectuarea diferenței simetrice a celor două mulțimi :

```
560 PRINT "DIFERENTA SIMETRICA : " : PRINT "G = {";
```

Se afişează elementele mulţimii D = A - B :

```
570 FOR I=1 TO L : PRINT D(I); : NEXT I
```

Se determină mulţimea D = B - A :

```
580 L=0
590 FOR I=1 TO M : K=0
600 FOR J=1 TO N
610 IF B(I)=A(J) THEN K=1
620 NEXT J
630 IF K<>0 THEN GOTO 650
640 L=L+1 : D(L)=B(I)
650 NEXT I
```

Se afişează elementele mulţimii D = B - A :

```
660 FOR I =1 TO L : PRINT D(I); : NEXT I
670 PRINT"}" : PRINT
```

1.2. Se consideră un şir *A* de numere cu *n* elemente distincte două cîte două. Să se facă un program BASIC prin care să se ordoneze acest şir în ordine crescătoare (descrescătoare) şi să se determine locul fiecărui element în şirul iniţial şi în şirul astfel ordonat.

Algoritmul este şi acum simplu dacă se ordonează şirul crescător, respectiv descrescător. Realizarea programului este uşurată de faptul că elementele şirului nu se repetă. Problema este rezolvată numai în cazul şirului ordonat crescător, celălalt caz rămînînd un exerciţiu pentru cititor.

Se şterge ecranul, zona de date, se introduce numărul de elemente ale mulţimii A şi elementele mulţimii A :

```
10 CLS : CLEAR
20 INPUT "Introduceti numarul de elemente : "; N : DIM
A(N)
```

Se introduc elementele şirului A şi se verifică dacă acestea sînt distincte două cîte două. De asemenea se păstrează elementele şirului A într-un nou şir B :

```
30 FOR I=1 TO N
40 PRINT"A("; I ;")="; : INPUT A(I)
50 FOR J=1 TO I - 1
60 IF A(I)<>A(J) THEN GOTO 80
70 PRINT "elementul se repeta" : GOTO 40
80 NEXT J
90 LET B(I)=A(I)
100 NEXT I
```

Se ordonează crescător elementele şirului A; pentru aceasta se foloseşte instrucţiunea SWAP X, Y care schimbă între ele valorile variabilelor X şi Y. În cazul utilizării unui calculator HC - 85, 88, 90, 91, sau orice alt calculator compatibil SPECTRUM, se foloseşte secvenţa : T=X, X=Y, Y=T. A nu se uita că, în cazul acestor calculatoare, instrucţiunea de atribuire LET este obligatorie, ceea ce nu se întîmplă în cazul limbajului GWBASIC folosit de calculatoarele care folosesc acest limbaj cum ar fi cele compatibile IBM - PC.

```
110 FOR I=1 TO N - 1
120 IF A(I) <= A(I + 1) THEN GOTO 140
130 SWAP A(I), A(I+1) : GOTO 110
140 NEXT I
```

Se afişează valorile şirului iniţial care au fost păstrate în şirul B :

```
150 PRINT "sir initial : " : PRINT "A = {"; : FOR I=1 TO N :
PRINT B(I); : NEXT I : PRINT"}"
```

Se afişează valorile şirului ordonat :

```
160 PRINT "sir ordonat : " : PRINT "A = {"; : FOR I=1 TO
N : PRINT A(I); : NEXT I : PRINT"}"
```

Se determină şi se afişează poziţia fiecărui element din şirul iniţial în şirul ordonat crescător :

```
170 FOR I=1 TO N
180 FOR J=1 TO N
190 IF NOT(B(I)=A(J)) GOTO 210
200 PRINT "el.a("; I ; ") ocupa pozitia "; J ; " in sirul ordonat"
210 NEXT J, I
```

1.3. Acelaşi text ca la problema 1.2., însă şirul poate avea elemente care se pot repeta. În acest caz să se indice un şir în care elementele apar o singură dată şi să se indice, pentru fiecare element al acestui şir, toate poziţiile în care apare acesta în şirul iniţial şi în şirul ordonat crescător, respectiv descrescător.

Să se rezolve problema poziţiilor elementului în şirurile ordonate şi fără a face ordonarea şirurilor.

Asemănătoare cu problema precedentă, apare cazul în care elementele şirului considerat se pot repeta. Rezolvarea acestei probleme se face în două moduri : primul prin care se ordonează mai întîi elementele mulţimii, iar al doilea în care elementele mulţimii nu mai sînt ordonate. Problema este rezolvată numai în cazul şirului ordonat crescător, celălat caz rămînînd un exrciţiu pentru cititor.

Primul program :

Se şterge ecranul şi zona de variabile. Se citeşte numărul de elemente ale mulţimii A şi se dimensionează două variabile de tip tablou unidimensional. Menţionăm că, în cazul calculatoarelor compatibile SPECTRUM, trebuie dată cîte o instrucţiune DIM pentru fiecare tablou în parte.

```
10 CLS : CLEAR
20 INPUT "Introduceti numarul de elemente : "; N : DIM
A(N),B(N)
```

Se introduc elementele şirului A care sînt copiate în B :

```
30 FOR I=1 TO N : PRINT"A("; I ;")="; : INPUT A(I) : B(I)=A(I)
: NEXT I
```

Se elimină din A elementele care se repetă :

```
40 L=1 : A(L)=B(1)
50 FOR I=1 TO N : K=0
60 FOR J=1 TO L
70 IF B(I)=A(J) THEN K=1
80 NEXT J
90 IF K<>0 THEN GOTO 110
100 L=L+1 : A(L)=B(I)
110 NEXT I
```

Se ordonează crescător elementele lui A :

```
120 FOR I=1 TO L - 1
130 IF A(I) <= A(I + 1) THEN GOTO 150
140 SWAP A(I), A(I+1) : GOTO 120
```

```
150 NEXT I
```

Se afişează elementele şirului iniţial :

```
160 PRINT "sir initial : " : PRINT"A = {";
170 FOR I=1 TO N : PRINT B(I); : NEXT I : PRINT "}"
```

Se afişează elementele şirului ordonat care conţine elementele şirului iniţial o singură dată :

```
180 PRINT "sir ordonat care contine elementele lui A o
singura data : " : PRINT "A = {";
190 FOR I=1 TO L : PRINT A(I); : NEXT I : PRINT "}"
```

Se determină şi se afişează toate poziţiile unui element din şirul iniţial în şirul ordonat :

```
200 FOR J=1 TO L
210 FOR I=1 TO N
220 IF B(I)<>A(J) GOTO 240
230 PRINT "elementul "; A(I); " ocupa pozitia "; J ; " in sirul
ordonat"
240 NEXT I, J
```

Al doilea program :

Se şterge ecranul, se iniţializează zona de date, se introduce numărul de elemente ale şirului şi se dimensionează variabila de tip tablou unidimensional, A :

```
10 CLS : CLEAR
20 INPUT "numarul elementelor sirului : "; N : DIM A(N)
```

Se indroduc și se afișează elementele șirului inițial :

```
30 FOR I = 1 TO N : PRINT "A("; I ;")="; : INPUT A(I) : NEXT
I
40 CLS : PRINT "sirul initial : " : PRINT"{";
50 FOR I=1 TO N : PRINT A(I); : NEXT I : PRINT"}"
```

Se determină cîte elemente sînt mai mici decît un element fixat și se afișează poziția fiecărui element în șirul ordonat :

```
60 FOR I=1 TO N : S=1
70 FOR J=1 TO N
80 IF A(I)>A(J) THEN S=S+1
90 NEXT J
100 PRINT "elementul a("; I ;") ocupa pozitia "; S ;" in sirul
ordonat crescator"
110 NEXT I
120 END
```

1.4. Se consideră un șir *A* de numere cu *n* elemente. Să se facă un program BASIC prin care să se introducă un număr de la tastatură și să se determine dacă acesta se găsește în șirul dat și, în caz afirmativ, să se indice de cîte ori apare elementul respectiv și toate pozițiile acestuia în următoarele șiruri : cel inițial, cel ordonat crescător și cel ordonat descrescător, fără a face efectiv ordonarea.

Problema nu ridică dificutăți, ea fiind din clasa celor anterioare. Vă prezentăm un exemplu de program pentru această problemă :

Se șterge ecranul, se inițializează zona de variabile, se citește numărul de elemente ale șirului A și se dimensionează un tablou unidimensional A :

```
10 CLS : CLEAR
20 INPUT "numarul de elemente  : "; N : DIM A(N)
```

Se indroduc și se afișează elementele șirului inițial :

```
30 FOR I=1 TO N : PRINT "A("; I ;")"; : INPUT A(I) : NEXT I
40 PRINT "sir initial : " : PRINT "{";
50 FOR I=1 TO N : PRINT A(I); : NEXT I : PRINT "}"
```

Se introduce un număr de la tastatură :

```
60 PRINT "Introduceti valoarea de cautata : "; : INPUT K
```

Se cercetează dacă numărul introdus se găsește în șirul inițial, pe ce poziții și de câte ori; dacă acesta nu se găsește în șirul inițial, se va afișa că se găsește de 0 ori în șir :

```
70 PRINT "in sirul initial "; K ;" apare pe pozitiile : " : S=0
80 FOR I=1 TO N
90 IF K<>A(I) THEN GOTO 110
100 PRINT I;","; : S=S+1
110 NEXT I
120 PRINT "  deci  de "; S ; " ori"
```

Se cercetează dacă numărul introdus se găsește în șirul ordonat crescător, pe ce poziții și de câte ori; dacă acesta nu se găsește în șirul inițial, se va afișa că se găsește de 0 ori în șir :

```
130 S=0
140 T=0
150 PRINT "in sirul ordonat crescator "; K ; " apare pe
pozitiile : "
160 FOR I=1 TO N
```

```
170 IF A(I)<K THEN S=S+1
180 IF A(I)=K THEN T=T+1
190 NEXT I
200 IF T=0 THEN GOTO 220
210 FOR I=1 TO T : PRINT S+I;","; : NEXT I
220 PRINT "  deci de "; T ; " ori"
```

Se cercetează dacă numărul introdus se găsește în șirul ordonat descrescător, pe ce poziții și de cîte ori; dacă acesta nu se găsește în șirul inițial, se va afișa că se găsește de 0 ori în șir :

```
230 S=0
240 T=0
250 FOR I=1 TO N
260 IF A(I)>K THEN S=S+1
270 IF A(I)=K THEN T=T+1
280 NEXT I
290 PRINT "in sirul ordonat descrescator "; K ; " apare pe
pozitiile : "
300 IF T=0 THEN GOTO 320
310 FOR I=1 TO T : PRINT S+I; ","; : NEXT I
320 PRINT "  deci de "; T ; " ori"
```

1.5., 1.6., 1.7. Aceste probleme au un algoritm comun, diferențele nefiind esențale decît în cerințe. Algoritmul prezentat aici este foarte instructiv și el poate fi folosit cu succes în multe alte situații. Fiind mai dificil, îl vom explica pe larg, urmărind primul program pentru problema 1.5.

Liniile 10 - 20 sînt pentru inițializarea calculatorului și pentru introducerea și validarea numărului de elemente ale mulțimii A. Am ales ca valoare maximă 16 pentru n, deoarece timpul de execuție al programului crește exponențial cu valoarea lui n.

În linia 40 sînt introduse valorile șirului A. Acest sir este ordonat de instrucțiunile de la liniile 50 - 100.

Liniile 110 - 140 stabilesc, pe de o parte, cîte elemente distincte sînt în şirul considerat şi, pe de altă parte, de cîte ori apare fiecare element în şirul iniţial (valorile L şi $C(i), i=1,\ldots,L$).

Pînă acum nimic nou. Să considerăm o bază de numeraţie dată de numărul elementelor distincte din şirul considerat, adică L. Cifrele le vom nota convenţional prin litere mari, de la A la P cu convenţia că "A" < "B" < . . . <"P" , "A" +1="B" , "B" +1= "C ",..., "X" +1= "BA" ,etc., "X" fiind ultima cifră din baza respectivă de numeraţie, adunarea făcîndu-se după regulile obişnuite. În fond, "A" reprezintă 0, "B" reprezintă 1, "X" reprezintă L - 1.

Variabila $A$$ reprezintă cifrele posibile de la "A" la "P" . Din această variabilă se selectează primele L caractere reprezentînd cifrele mai sus amintite.

Variabila de tip şir $B$$ va reprezenta un număr în baza L şi cifrele acestui număr sînt în corespondenţă biunivocă cu termenii şirului ordonat, fiecărui element al şirului corespunzîndu-i o cifră şi numai una. De aceea, fiecare cifră va apare exact de atîtea ori în componenţa lui $B$$ de cîte ori apare elementul corespunzător în şirul A. De aceea $B$$ va avea exact n caractere.

Ideea este acum simplă. Vom considera toate numerele de la $B$$ pînă la "$BAAA\ldots A$" , cifra A fiind luată de n ori. Linia 150 defineşte variabila $A$$, iar linia 160 defineşte variabila $B$$. Se consideră acum o nouă variabilă $C$$ care are, pentru început, valoarea $B$$. Cu ajutorul subrutinei 210 se afişează valoarea lui $C$$. Subrutina 220 - 280 realizează adunarea cu 1 a variabilei $C$$. Pentru a vedea dacă cifrele lui $C$$ sînt aceleaşi cu cele ale lui $B$$ dar în altă ordine, se ordonează $C$$ într-o nouă variabilă $D$$. Această operaţie este realizată de subrutina 290 - 360. Dacă $D$$ coincide cu $B$$, atunci rezultatul este corect şi se afişează şirul corespunzător lui $C$$. În caz contrar, se continuă procedeul pînă cînd se ajunge la "$AAA...A$" , cînd se încheie programul. Dăm în continuare cele trei programe corespunzătoare celor trei probleme.

Problema 1.5.

Se consideră un şir A de numere cu n elemente. Să se determine toate anagramele acestui şir. (Prin anagramă înţelegem un nou şir

care are aceleaşi elemente cu şirul iniţial şi fiecare element apare exact de atîtea ori ca şi în şirul iniţial).

Se şterge ecranul, se iniţializează zona de variabile, se introduce numărul de elemente ale şirului, se validează dacă acest număr este natural, cuprins între 1 şi 17, exclusiv, şi se dimensionează două tablouri unidimensionale A şi C. Tabloul C va conţine numărul de repetiţii ale fiecărui element din A :

```
10 CLS : CLEAR : INPUT "Introduceti numarul de elemente
: "; N
20 IF N<2 OR N<>INT(N) OR N>16 THEN 10
30 DIM A(N),C(N)
```

Se introduc elementele şirului A şi se iniţializează valorile tabloului C cu 1. Din C vor fi considerate doar atîtea elemente cîte elemente conţine şirul A :

```
40 FOR I=1 TO N : PRINT "Introduceti elementul "; I ; " din
sir : "; : INPUT A(I) : C(I)=1 : NEXT I
```

Se ordonează elementele şirului A ; după cum se observă, această ordonare este realizată cu o variabilă de test S, spre deosebire de ordonările anterioare cînd nu am folosit această variabilă ; S ia valoarea 1 numai dacă, la o parcurgere a şirului, s-a efectuat o schimbare ; în caz contrar are valoarea 0 :

```
50 S=0
60 FOR I=1 TO N - 1
70 IF A(I)<=A(I+1) THEN 90
80 SWAP A(I), A(I+1) : S=1
90 NEXT I
100 IF S=1 THEN 50
```

element în şirul iniţial A ; numărul elementelor distincte este păstrat de variabila L, iar de cîte ori se găseşte un element în A, în tabloul C:

```
110 L=1 : FOR I=1 TO N - 1
120 IF A(I)=A(I+1) THEN C(L)=C(L)+1 : GOTO 140
130 L=L+1
140 NEXT I
```

Se construieşte o variabilă de tip caracter A$ care conţine cifrele în ordine crescătoare asociate oricărei baze de numeraţie < 17 :

```
150 A$="ABCDEFGHIJKLMNOP"
```

Se construieşte o variabilă de tip caracter B$ care reprezintă un număr în baza L şi care conţine pe primele poziţii cifra "A" de atîtea ori de cîte ori apare în şirul A cel mai mic element, pe următoarele poziţii cifra "B" de atîtea ori de cîte ori apare următorul element ordonat crescător în şirul A, etc. ; se observă folosirea funcţiei MID$(A$,I,J) care extrage un subşir al şirului A$ care începe din poziţia I şi este format din J caractere; pentru calculatoarele din familia HC şi compatibile, funcţia MID$(A$,I,J) este echivalentă cu A$(I TO I+J - 1), deci, pentru testarea acestui program şi ale altora care conţin această funcţie, este necesar să se facă modificarea corespunzătoare :

```
160 B$="" : FOR I=1 TO L : FOR J=1 TO C(I) : B$=B$+
MID$(A$, I, 1) : NEXT J, I
```

Pentru a nu altera, pe parcursul execuţiei programului, variabila B$, o copiem în variabila C$ cu care vom lucra şi, deoarece aceasta corespunde unei anagrame a lui B$, vom afişa anagrama corespunzătoare şirului iniţial A (GOSUB 210).

```
170 C$=B$ : GOSUB 210
```

Se determină succesorul lui C$ în raport cu adunarea ; dacă din

acest succesor reținem numai ultimele n caractere, n fiind numărul de elemente ale șirului inițial, operația succesor devine ciclică, astfel încît ne vom opri atunci cînd C$ conține numai cifra "A" : observăm utilizarea funcției LEFT$(A$,K) care extrage subșirul format din primele K caractere ale șirului A$; pentru calculatoarele compatibile HC,. această funcție va fi înlocuită cu A$(TO K) :

```
180 GOSUB 220 : IF C$=LEFT$("AAAAAAAAAAAAAAAA",
N) THEN END
```

Variabila C$ este ordonată în variabila D$ care trebuie să coincidă cu B$ dacă C$ reprezintă o anagramă a lui B$; GOSUB 290 realizează tocmai trimiterea la subrutina de ordonare pentru a obține D$; în cazul în care D$=B$, avem de-a face cu o anagramă, deci apelăm subrutina de afișare (GOSUB 210) ; indiferent de rezultat, mergem la 180 pentru repetarea procedeului cu noua valoare a lui C$; cu linia 200 se încheie și programul principal :

```
190 GOSUB 290 : IF D$=B$ THEN GOSUB 210
200 GOTO 180
```

Aceasta este o subrutină de afișare a unei anagrame a șirului inițial A corespunzătoare unei valori a lui C$; despre funcția MID$ am vorbit anterior ; dar mai apare o funcție, INSTR(A$,B$), care are valoarea 0 dacă B$ nu este subșir în A$ și are valoarea k dacă B$ este subșir în A$ și prima apariție a subșirului B$ în șirul A$ este pe poziția k ; pe calculatoarele HC sau compatibile, această funcție poate fi doar simulată printr-o subrutină astfel :

```
210 for i=1 to n : gosub 500 : print a(r), : next i : print : return
.....
500 let r=0 : for u=1 to n
510 if b$(u)=c$(i) and r=0 then let r=u
520 next u : return
```

După cum ne aducem aminte, B$ reprezintă o imagine a șirului ordonat A, iar C$ reprezintă tocmai o imagine a unei anagrame a lui A. Funcția INSTR(B$,MID$(C$,I,1)) realizează tocmai această permutare a indicilor :

```
210 FOR I=1 TO N : PRINT A(INSTR(B$,MID$(C$,I,1))), :
NEXT I : PRINT : RETURN
```

Următoarea subrutină realizează calculul succesorului numărului reprezentat de C$. Deoarece caracterele care reprezintă cifrele sînt consecutive din punct de vedere al codului ASCII, succesorul unui caracter va fi caracterul cu codul ASCII corespunzător. Astfel, dacă caracterul ales este "C" care are codul ASCII 67, succesorul va fi caracterul cu codul 68, adică "D". Ultima cifră a lui C$ va fi cea de pe poziția LEN(C$) care reprezintă lungimea lui C$. După cum se face adunarea în mod obișnuit, vom începe cu această cifră :

```
220 I=LEN(C$)
```

Cifra de pe locul I este dată de MID$(C$,I,1), respectiv C$(I) pentru calculatoarele compatibile HC. Codul ASCII al acestei cifre este dat de funcția ASC, echivalentă cu funcția CODE pentru calculatoarele compatibile HC. Succesorul va avea codul ASCII dat de ASC(MID$(C$,I,1))+1, iar caracterul corespunzător este dat de U$ din linia 230 :

```
230 U$=CHR$(ASC(MID$(C$,I,1))+1)
```

S-ar putea ca acest caracter să depășească poziția L din A$, astfel încît, în acest caz, caracterul de pe poziția I din C$ trebuie să fie înlocuit cu cea mai mică cifră , adică cu "A". Deci, dacă U$ este una din cifre , se înlocuiește caracterul de pe poziția I din C$ cu U$ și subrutina se încheie. În caz contrar, caracterul de pe poziția I din C$ devine A și ne rămîne un rest , astfel că trebuie să modificăm cifra de pe locul I -1, procedeul fiind continuat în mod analog:

```
240 IF U$<MID$(A$,L+1,1) THEN GOTO 270
250 MID$(C$,I,1)="A" : I=I - 1 : IF I=0 THEN 280
260 GOTO 230
270 MID$(C$,I,1)=U$
```

```
260 GOTO 230
270 MID$(C$,I,1)=U$
280 RETURN
```

Această subrutină realizează ordonarea caracterelor şirului D$. Putem folosi, fără restricţii, operatorii <, >, <=, >=, <>, =, deoarece compararea se face întrelungimile şi codurile ASCII ale caracterelor :

```
290 D$=C$
300 S=0
310 FOR I=1 TO N - 1
320 IF MID$(D$,I,1)<=MID$(D$,I+1,1) THEN 340
330 SWAP MID$(D$,I,1), MID$(D$,I+1,1) : S=1
340 NEXT I
350 IF S=1 THEN 300
360 RETURN
```

Problema 1.6.

Se consideră o mulţime *A* de numere cu *n* elemente. Să se construiască un şir care să conţină pe primele *k* poziţii (*k*<*n*) elemente ale mulţimii *A* ordonate crescător, iar pe următoarele *n* - *k* poziţii, celelalte elemente ordonate descrescător. Se vor indica toate posibilităţile.

Pentru programele 1.6. şi 1.7. nu vom mai da multe explicaţii, deoarece ar trebui să repetăm multe din cele afirmate la programul 1.5.

```
10 CLS : CLEAR : INPUT "Introduceti numarul elementelor
din sir : "; N
20 IF N<2 OR N<>INT(N) OR N>16 THEN 10
```

Se introduce numărul de elemente pe care dorim să-l selectăm și se validează acesta :

```
30 INPUT "Introduceti k  : ";G
40 IF G<2 OR G<>INT(G) OR G>N THEN 10
50 DIM A(N), H(N)
60 FOR I=1 TO N : PRINT "Introduceti elementul "; I ; " din
sir  : "; : INPUT A(I) : NEXT I
70 S=0
80 FOR I=1 TO N - 1
90 IF A(I)<=A(I+1) THEN 110
100 R=A(I) : A(I)=A(I+1) : A(I+1)=R : S=1
110 NEXT I
120 IF S=1 THEN 70
130 E=0 : FOR I=1 TO N - 1
140 IF A(I)=A(I+1) THEN E=1
150 NEXT I
160 IF E=1 THEN 10
170 A$="ABCDEFGHIJKLMNOP"
180 B$=LEFT$(A$,N)
190 C$=LEFT$(A$,G) : GOSUB 230
200 GOSUB 310 : IF C$=LEFT$("AAAAAAAAAAAAAAAA",
G) THEN END
210 GOSUB 380 : GOSUB 460
220 GOTO 200
230 FOR I=1 TO G : H(I)= A(INSTR(A$,MID$(C$,I,1))) :
NEXT I
240 W=G : FOR I=1 TO N : V=0 : FOR J=1 TO G
250 IF A(I)=H(J) THEN V=1
260 NEXT J
```

```
270 IF V=0 THEN W=W+1 : H(W)=A(I)
280 NEXT I
290 FOR I=1 TO G : PRINT H(I), : NEXT I : FOR I=N TO
G+1 STEP -1 : PRINT H(I), : NEXT I
300 RETURN
310 I=LEN(C$)
320 U$=CHR$(ASC(MID$(C$,I,1))+1)
330 IF U$<MID$(A$,N+1,1) THEN GOTO 360
340 MID$(C$,I,1)="A" : I=I-1 : IF I=0 THEN 370
350 GOTO 320
360 MID$(C$,I,1)=U$
370 RETURN
380 D$=C$
390 S=0
400 FOR I=1 TO G-1
410 IF MID$(D$,I,1)<=MID$(D$,I+1,1) THEN 430
420 U$=MID$(D$,I,1) : MID$(D$,I,1)=MID$(D$,I+1,1) :
MID$ (D$,I+1,1)=U$ : S=1
430 NEXT I
440 IF S=1 THEN 390
450 RETURN
460 E=0 : FOR I=1 TO G-1
470 IF MID$(D$,I,1)=MID$(D$,I+1,1) THEN E=1
480 NEXT I
490 IF E=0 AND D$=C$ THEN GOSUB 230
500 RETURN
```

Problema 1.7.

Se consideră o mulţime A de numere cu n elemente şi $k<n$ un

număr natural. Să se determine toate submulțimile distincte de numere cu *k* elemente din *A*.

```
10 CLS : CLEAR : INPUT "Introduceti numarul de elemente
ale sirului : ", N
20 IF N<2 OR N<>INT(N) OR N>16 THEN 10
30 INPUT "Introduceti k : ", G
40 IF G<2 OR G<>INT(G) OR G>N THEN 10
50 DIM A(N)
60 FOR I=1 TO N : PRINT "Introduceti elementul "; I ; " din
sir : " : INPUT A(I) : NEXT I
70 S=0
80 FOR I=1 TO N - 1
90 IF A(I)<=A(I+1) THEN 110
100 R=A(I) : A(I)=A(I+1) : A(I+1)=R : S=1
110 NEXT I
120 IF S=1 THEN 70
130 E=0 : FOR I=1 TO N-1
140 IF A(I)=A(I+1) THEN E=1
150 NEXT I
160 IF E=1 THEN 10
170 A$="ABCDEFGHIJKLMNOP"
180 B$=LEFT$(A$,N)
190 C$=LEFT$(A$,G) : GOSUB 230
200 GOSUB 240 : IF C$=LEFT$("AAAAAAAAAAAAAAAA",
G) THEN END
210 GOSUB 310 : GOSUB 390
220 GOTO 200
230 FOR I=1 TO G : PRINT A(INSTR(A$,MID$(C$,I,1))), :
NEXT I : PRINT : RETURN
240 I=LEN(C$)
```

```
250 U$=CHR$(ASC(MID$(C$,I,1))+1)
260 IF U$<MID$(A$,N+1,1) THEN GOTO 290
270 MID$(C$,I,1)="A" : I=I-1 : IF I=0 THEN 300
280 GOTO 250
290 MID$(C$,I,1)=U$
300 RETURN
310 D$=C$
320 S=0
330 FOR I=1 TO G - 1
340 IF MID$(D$,I,1)<=MID$(D$,I+1,1) THEN 360
350 U$=MID$(D$,I,1) : MID$(D$,I,1)=MID$(D$,I+1,1) :
MID$(D$,I+1,1)=U$ : S=1
360 NEXT I
370 IF S=1 THEN 320
380 RETURN
390 E=0 : FOR I=1 TO G - 1
400 IF MID$(D$,I,1)=MID$(D$,I+1,1) THEN E=1
410 NEXT I
420 IF E=0 AND D$=C$ THEN GOSUB 230
430 RETURN
```

1.8. Fie *A* un şir de numere cu *n* elemente şi *k*<*n* un număr natural. Să se construiască un nou şir care să conţină, pe primele k locuri, ultimele *k* elemente ale şirului iniţial, iar pe ultimele *n* - *k* locuri, primele *n* - *k* elemente ale şirului iniţial.

Această problemă, foarte simplă, este un exerciţiu util pentru începători, de aceea nu vom prezenta decît programul.

```
10 CLS : CLEAR
20 INPUT "Numarul elementelor din sir : "; N
30 IF N<2 OR N<>INT(N) GOTO 10
```

```
40 DIM A(N) : DIM B(N)
50 FOR I=1 TO N
60 PRINT "Introduceti elementul "; I ; " din sir : ";
70 INPUT A(I)
80 NEXT I
90 INPUT "Introduceti k : "; K
100 IF K>N THEN GOTO 80
110 LET C=1
120 FOR I=K+1 TO N
130 LET B(C)=A(I) : LET C=C+1
140 NEXT I
150 FOR I=1 TO K
160 LET B(C)=A(I) : LET C=C+1
170 NEXT I
180 PRINT "Sirul obtinut : ";
190 FOR I=1 TO N
200 PRINT B(I); " ";
210 NEXT I
```

1.9. Fie *A* un şir de n numere.

a) Să se calculeze suma termenilor acestui şir ;

b) să se calculeze produsul termenilor acestui şir ;

c) să se determine dacă termenii şirului sînt în progresie aritmetică ;

d) să se determine dacă termenii acestui şir sînt în progresie geometrică ;

e) să se calculeze media armonică a elementelor acestui şir;

La punctele c) şi d), în caz că răspunsul este afirmativ, se va specifica raţia.

Programul pentru rezolvarea problemei este următorul :

```
10 CLS : CLEAR
```

```
20 INPUT "Introduceti numarul elementelor din sir : "; N
30 IF N=0 OR N<>INT(N) THEN 20
40 DIM A(N)
50 IF N=0 THEN GOTO 20
60 FOR I=1 TO N
70 PRINT "Introduceti elementul "; I ; " din sir : ";
80 INPUT A(I)
90 NEXT I
100 LET S=0
110 LET P=1
120 FOR I=1 TO N
130 LET P=P*A(I)
140 LET S=S+A(I)
150 NEXT I
160 PRINT "Suma elementelor din sir este : "; S
170 PRINT "Produsul elementelor din sir este : "; P
180 LET C=A(2) - A(1)
190 FOR I=1 TO N - 1
200 IF C=A(I+1) - A(I) THEN GOTO 220
210 IF C<>A(I+1) - A(I) THEN PRINT "Termenii acestui sir
nu sint in progresie aritmetica." : GOTO 240
220 NEXT I
230 PRINT "Termenii acestui sir sint in progresie aritmetica,
iar ratia este : ";C
235 IF A(1)=0 THEN 270
240 LET Q=A(2)/A(1)
250 FOR I=1 TO N - 1
255 IF A(I)=0 THEN 270
260 IF Q=A(I+1)/A(I) THEN GOTO 280
```

```
270 IF Q<>A(I+1)/A(I) THEN PRINT "Termenii acestui sir
nu sint in progresie geometrica." : GOTO 310
280 NEXT I
290 PRINT "Termenii acestui sir sint in progresie
geometrica, iar ratia este : "; Q
300 LET K=0
310 FOR I=1 TO N
320 LET K=K+1/A(I)
330 NEXT I
340 LET O=N/K
350 PRINT "Media armonica este : "; O
```

1.10. Se consideră un şir A de numere cu n elemente şi un număr natural $k < n$. Să se găsească elementul de pe poziţia k în şirul ordonat crescător (descrescător), fără a ordona şirul.

```
10 CLS : CLEAR
20 INPUT "Introduceti numarul elementelor din sir : "; N
30 DIM A(N)
40 IF N=0 THEN GOTO 10
50 FOR I=1 TO N
60 PRINT "introduceti elementul "; I ; " din sir : ";
70 INPUT A(I)
80 NEXT I
90 INPUT "Pozitia elementului pentru care vreti sa vedeti
ce pozitie ocupa in sirul ordonat : "; K
100 IF K<1 OR K>N THEN GOTO 90
110 LET S=1
120 FOR I=1 TO N
130 IF A(I)<A(K) THEN LET S=S+1
140 NEXT I
```

```
150 PRINT "Elementul "; A(K); " de pe pozitia "; K ; " in sirul
neordonat, ocupa pozitia "; S ; " in sirul ordonat."
```

1.11. Se consideră un şir *A* de numere cu *n* elemente. Să se determine elementul maxim (minim) din şir şi toate poziţiile în care acesta apare printr-o singură parcurgere completă a şirului.

Programul pentru rezolvarea problemei :

```
10 INPUT "Introduceti numarul de elemente ale sirului : ",
N
20 DIM A(N)
30 DIM B(N)
40 FOR I=1 TO N
50 PRINT "Introduceti elementul "; I ; " din sir : " : INPUT
A(I)
60 B(I)=0
70 NEXT I
80 MAX=A(1)
90 K=1
100 B(K)=1
110 FOR I=2 TO N
120 IF MAX<A(I) THEN GOTO 180
130 IF MAX=A(I) THEN GOTO 220
140 GOTO 240
150 FOR J=1 TO N
160 B(J)=0
170 NEXT J
180 K=1
190 B(K)=I
200 MAX=A(I)
210 GOTO 240
```

```
220 K=K+1
230 B(K)=I
240 NEXT I
250 PRINT "Elementul maxim este : ", MAX
260 PRINT "El apare pe pozitiile : "
270 FOR I=1 TO N
280 IF B(I)<>0 THEN PRINT B(I);
290 NEXT I
300 END
```

1.12. Se consideră un şir de numere *A* cu *n* elemente. Să se determine:

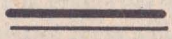
a) cîte elemente sînt pozitive, negative şi cîte sînt nule ;

b) cîte sînt întregi şi cîte nu.

Programul pentru rezolvarea problemei :

```
10 CLS : CLEAR
20 INPUT "Introduceti numarul elementelor : "; N
30 IF N=0 THEN GOTO 20
40 DIM A(N)
50 FOR I=1 TO N
60 PRINT "Introduceti elementul "; I ; " din sir : ";
70 INPUT A(I)
80 NEXT I
90 LET C=0
100 LET P=0
110 LET K=0
120 FOR I=1 TO N
130 PRINT "a("; I ; ")="; A(I)
140 NEXT I
150 FOR I=1 TO N
```

```
160 IF A(I)<0 THEN LET C=C+1
170 IF A(I)>0 THEN LET P=P+1
180 IF A(I)=0 THEN LET K=K+1
190 NEXT I
200 PRINT "In acest sir avem : "
210 PRINT "      -"; C ; " element(e) negativ(e);"
220 PRINT "      -"; P ; " element(e) pozitiv(e);"
230 PRINT "      -"; K ; " element(e) egal(e) cu 0;"
240 LET T=0
250 LET Q=0
260 FOR I=1 TO N
270 IF A(I)=INT(A(I)) THEN LET T=T+1
280 IF A(I)<>INT(A(I)) THEN LET Q=Q+1
290 NEXT I
300 PRINT "      -"; T ; " element(e) intreg(i);"
310 PRINT "      -"; Q ; " element(e) care nu sint(este)
intreg(i)."
320 END
```



2.2. PROBLEME CU ŞIRURI DE CARACTERE.

Multe dintre problemele acestui capitol sînt simple, ele fiind puse aici doar pentru formarea deprinderilor de a lucra cu şiruri de caractere. Pentru lămurirea cititorului, la problemele mai dificile, vom interveni cu explicaţii. Referitor la utilizarea programelor pe calculatoarele HC şi compatibile au fost date explicaţii în secţiunea precedentă. În cazul în care vor mai apare şi alte neconcordanţe, se vor face referirile necesare la momentul potrivit.

2.1. Se consideră un şir de caractere introdus de la tastatură. Să se determine lungimea sa şi codurile ASCII ale caracterelor şirului.

```
10 INPUT "Introduceti sirul de caractere N : "; C$
20 L=LEN(C$)
30 PRINT "Sirul are "; L ;" caractere"
40 PRINT "Caracterele ASCII corespunzatoare sint : "
50 FOR I=1 TO L
60 A$=MID$(C$,I,1)
70 PRINT A$, ASC(A$)
80 NEXT I
90 END
```

2.2. Se consideră un şir de caractere şi *k* un număr natural. Să se cerceteze dacă *k* este mai mic decît lungimea şirului şi, în caz afirmativ, să se afişeze cele *k* elemente din mijloc.

```
10 INPUT "Introduceti sirul de caractere:", C$
20 L=LEN(C$)
30 PRINT "Sirul are "; L ; " caractere"
40 INPUT "Introduceti numarul de caractere de afisat:",K
50 IF K=0 OR K>L THEN GOTO 40
60 PRINT "Subsirul extras este:"
70 LET P=INT((L - K)/2)+1
```

```
80 PRINT MID$(C$,P,K)
90 END
```

2.3. Cum putem introduce caracterele <CR> şi <"> de la tastatură în cadrul unui şir de caractere ?

Necesitatea ca într-un program să avem nevoie de introducerea de la tastatură a caracterelor <CR> sau <"> este ceva mai rară. Prezentăm, în continuare, modalităţile utilizate pentru introducerea unui caracter sau a unui şir de caractere, indiferent de codul ASCII al fiecărui caracter din şir, utilizînd atît funcţia INKEY$ cît şi funcţia INPUT$(*n*).

Prima metodă rezolvă problema introducerii unui singur caracter de la tastatură şi afişarea codului ASCII al caracterului, deoarece nu toate caracterele sînt vizibile pe ecran ; variabila A$ este mai întîi iniţializată cu şirul vid; acest lucru este necesar, deoarece, în caz contrar, este posibil ca variabila A$ să conţină caractere dacă ea a fost utilizată anterior în program ; se aşteaptă un caracter de la tastatură ; dacă acesta nu a fost tastat, adică A$ este tot şirul vid, se reia linia 10 de la început ; în momentul tastării se afişează codul ASCII al caracterului (linia 20). De menţionat că, în acest caz, caracterele introduse astfel sînt ascunse , adică ele nu sînt afişate pe ecran. De asemenea, în acest mod, poate fi introdus orice caracter care are codul ASCII cuprins între 1 şi 255, dacă pe calculator este prevăzută toată gama de caractere. La sfîrşitul cărţii sînt prezentate codurile ASCII pentru calculatoarele compatibile IBM - PC şi pentru cele din familia HC.

```
10 A$="" : A$=INKEY$ : IF A$="" THEN 10
20 PRINT ASC(A$)
```

Al doilea program utilizează secvenţa de mai sus pentru introducerea unui şir de caractere. Deoarece nu ne putem opri dacă nu am specificat o condiţie, am indicat, drept sfîrşit al şirului, ultimul caracter introdus înainte de apăsarea tastei <ESC> care are codul ASCII 27 (la calculatoarele din familia HC acest caracter nu există, dar el poate fi înlocuit cu oricare altul).

```
10 A$=""
20 X$="" : X$=INKEY$ : IF X$="" THEN 20
30 IF ASC(X$)<>27 THEN A$=A$+X$ : GOTO 20
50 FOR I=1 TO LEN(A$) : PRINT ASC(MID$(A$,I,1)), : NEXT
I
```

Secvența următoare poate fi utilizată numai pentru calculatoarele care folosesc GWBASIC dar, în nici un caz, pe cele din familia HC. Pentru acestea se poate crea o secvență de program utilizînd ideile de mai sus. Este folosită funcția INPUT$(n) care arată că trebuie să introducem exact n caractere pentru variabila A$. Și în acest caz caracterele nu sînt afișate pe ecran.

```
10 A$=INPUT$(10)
20 FOR I=1 TO LEN(A$) : PRINT ASC(MID$(A$,I,1)), :
NEXT I
```

Simularea secvenței anterioare pe calculatoarele din familia HC se poate face ca în următorul program :

```
10 A$=""
20 FOR I=1 TO 10
30 A$=A$+INKEY$
40 NEXT I
50 FOR I=1 TO LEN(A$) : PRINT CODE(A$(I)), : NEXT I
```

2.4. Să se determine numărul de litere mari, litere mici, cifre și caractere speciale dintr-un șir de caractere.

```
10 LET S$="'-=[];',./\~!@#$%^ &*()_+{}:"+CHR$(34)+
"<>?|"
20 LET T$="abcdefghijklmnopqrstuvwxyz"
```

```
30 LET N$="0123456789"
40 LET  P$="ABCDEFGHIJKLMNOPQRSTUVXYZ"
50 DIM L(LEN(T$))
60 DIM C(LEN(N$))
70 DIM Q(LEN(S$))
80 DIM K(LEN(P$))
90 FOR I=1 TO LEN(T$)
100 LET L(I)=0
110 NEXT I
120 FOR I=1 TO LEN(N$)
130 LET C(I)=0
140 NEXT I
150 FOR I=1 TO LEN(S$)
160 LET Q(I)=0
170 NEXT I
180 FOR I=1 TO LEN(P$)
190 LET K(I)=0
200 NEXT I
210 INPUT "Introduceti textul :", F$
220 FOR I=1 TO LEN(F$)
230 FOR J=1 TO LEN(T$)
240 IF MID$(F$,I,1)=MID$(T$,J,1) THEN L(J)=L(J)+1
250 NEXT J
260 FOR J=1 TO LEN(N$)
270 IF MID$(F$,I,1)=MID$(N$,J,1) THEN C(J)=C(J)+1
280 NEXT J
290 FOR J=1 TO LEN(S$)
300 IF MID$(F$,I,1)=MID$(S$,J,1) THEN Q(J)=Q(J)+1
310 NEXT J
320 FOR J=1 TO LEN(P$)
```

```
330 IF MID$(F$,I,1)=MID$(P$,J,1) THEN K(J)=K(J)+1
340 NEXT J
350 NEXT I
360 PRINT "Litere mici intilnite in text : "
370 FOR I=1 TO LEN(T$)
380 IF L(I)>0 THEN PRINT MID$(T$,I,1); "-"; L(I); "  ";
390 NEXT I
400 PRINT : PRINT "Cifre intilnite in text : "
410 FOR I=1 TO LEN(N$)
420 IF C(I)>0 THEN PRINT MID$(N$,I,1); "-"; C(I);"  ";
430 NEXT I
440 PRINT : PRINT "Litere mari intilnite in text : "
450 FOR I=1 TO LEN(P$)
460 IF K(I)>0 THEN PRINT MID$(P$,I,1); "-"; K(I); " ";
470 NEXT I
480 PRINT : PRINT "Caractere speciale intilnite in text : "
490 FOR I=1 TO LEN(S$)
500 IF Q(I)>0 THEN PRINT MID$(S$,I,1); "-"; Q(I); "  ";
510 NEXT I
520 END
```

2.5. Să se determine dacă un caracter introdus de la tastatură se găsește într-un șir și, apoi, să se determine toate pozițiile în care se găsește acesta în șirul considerat.

```
10 INPUT "Introduceti sirul de caractere : ", N$
20 L=LEN(N$)
30 INPUT "Introduceti caracterul de cautat : ", C$
40 S=0
50 FOR I=1 TO L
60 IF MID$(N$,I,1)=C$  THEN S=S+1
```

```
70 NEXT I
80 PRINT "Caracterul "; C$; " apare in sirul initial de ";S;"
ori"
90 IF S=0 THEN GOTO 140
100 PRINT "pe pozitiile : "
110 FOR I=1 TO L
120 IF MID$(N$,I,1)=C$ THEN PRINT I;
130 NEXT I
140 END
```

2.6. Să se determine dacă un şir de caractere este subşir al altui şir de caractere şi să se precizeze poziţia de început.

Această problemă se rezolvă imediat pe calculatoarele compatibile IBM - PC, utilizînd funcţia INSTR(A$,B$) despre care am mai vorbit şi care are valoarea 0, dacă B$ nu este subşir în A$ şi ia valoarea k>0, dacă B$ este subşir în A$ şi apare prima oară în A$ pe poziţia k. Pe calculatoarele din familia HC prezentăm o simulare completă a acestei funcţii :

```
10 INPUT "Introduceti sirul de caractere : ", N$
20 INPUT "Introduceti subsirul de cautat : ", M$
30 IF LEN(N$)>LEN(M$) THEN GOTO 60
40 PRINT "Eroare - reia !"
50 GOTO 20
60 FOR I=1 TO LEN(N$)
70 LET K=0
80 IF MID$(N$,I,1)<>MID$(M$,1,1) THEN GOTO 140
90 FOR J=1 TO LEN(M$) - 1
100 IF MID$(N$,I+J,1)<>MID$(M$,J+1,1) THEN LET K=1
110 NEXT J
120 IF K<>0 THEN GOTO 140
```

```
130 PRINT "Subsirul '"; M$; "' apare in sirul initial incepind
cu pozitia"; I
140 NEXT I
150 END
```

2.7. Prin *cuvînt* înţelegem o succesiune de litere, din care numai prima literă poate fi majusculă şi care nu conţine spaţii sau caractere speciale. Să se extragă toate cuvintele dintr-un şir de caractere dat.

```
10 CLS : CLEAR : C$=""
20 INPUT "Introduceti textul : ", N$
30 PRINT "Cuvintele aparute in text sint : "
40 FOR I=1 TO LEN(N$)
50 A$=MID$(N$,I,1)
60 IF ASC(A$)<91 AND ASC(A$)>64 AND C$<>"" THEN
GOTO 110
70 IF (ASC(A$)<91 AND ASC(A$)>64 AND C$="") OR
(ASC (A$)>96 AND ASC(A$)<123) THEN GOTO 140
80 IF C$<>"" THEN PRINT C$
90 C$=""
100 GOTO 160
110 PRINT C$
120 C$=A$
130 GOTO 160
140 C$=C$+A$
150 GOTO 160
160 NEXT I
170 IF C$<>"" THEN PRINT C$
180 END
```

2.8. Să se anagrameze un cuvînt care conţine numai litere mari.

Problema este asemănătoare cu 1.5., singurele deosebiri constînd din introducerea datelor şi subrutina de afişare.

Se şterge ecranul, se iniţializează zona de variabile, se introduce X$, se iniţializează un tablou unidimensional C care conţine numărul de repetări ale caracterelor care intervin în X$:

```
10 CLS : CLEAR
20 INPUT "Introduceti sirul"; X$ : DIM C(LEN(X$)) : FOR
I=1 TO LEN(X$) : C(I)=1 : NEXT I
```

Se validează şi se ordonează alfabetic caracterele lui X$:

```
30 S=0
40 E=0 : FOR I=1 TO LEN(X$) - 1
50 IF MID$(X$,I,1)<"A" OR MID$(X$,I,1)>"Z" THEN E=1
60 IF MID$(X$,I,1)<=MID$(X$,I+1,1) THEN 80
70 U$=MID$(X$,I,1) : MID$(X$,I,1)=MID$(X$,I+1,1) : MID$
(X$, I+1,1)=U$ : S=1
80 NEXT I
90 IF E=1 THEN 10
100 IF S=1 THEN 30
```

De aici şi pînă la sfîrşitul programului principal nu există nici o diferenţă esenţială faţă de programul 1.5. :

```
110 L=1 : FOR I=1 TO LEN(X$) - 1
120 IF MID$(X$,I,1)=MID$(X$,I+1,1) THEN C(L)=C(L)+1 :
GOTO 140
130 L=L+1
140 NEXT I
150 A$="ABCDEFGHIJKLMNOP"
```

```
160 B$="" : FOR I=1 TO L : FOR J=1 TO C(I) : B$=B$+
MID$ (A$,I,1) : NEXT J, I
170 C$=B$ : GOSUB 210
180 GOSUB 220 : IF C$=LEFT$("AAAAAAAAAAAAAAAA",
LEN (X$)) THEN END
190 GOSUB 290 : IF D$=B$ THEN GOSUB 210
200 GOTO 180
```

Această subrutină, bazată pe aceeaşi idee ca şi la problema 1.5., stabileşte corespondenţa dintre poziţia caracterelor lui X$ şi cele corespunzătoare anagramei generată de C$:

```
210 FOR I=1 TO LEN(X$) : PRINT MID$(X$, INSTR(B$,
MID$(C$, I,1)),1); : NEXT I : PRINT : RETURN
```

Subrutinele 220 - 280 şi 290 - 360 sînt identice cu cele de la programul 1.5.

```
220 I=LEN(C$)
230 U$=CHR$(ASC(MID$(C$,I,1))+1)
240 IF U$<MID$(A$,L+1,1) THEN GOTO 270
250 MID$(C$,I,1)="A" : I=I-1 : IF I=0 THEN 280
260 GOTO 230
270 MID$(C$,I,1)=U$
280 RETURN

290 D$=C$
300 S=0
310 FOR I=1 TO LEN(X$)-1
320 IF MID$(D$,I,1)<=MID$(D$,I+1,1) THEN 340
```

```
330 U$=MID$(D$,I,1) : MID$(D$,I,1)=MID$(D$,I+1,1) :
MID$(D$,I+1,1)=U$ : S=1
340 NEXT I
350 IF S=1 THEN 300
360 RETURN
```

2.9. Să se transforme toate literele mici ale unui text în litere mari.

```
10 INPUT "Introduceti textul :",T$
20 PRINT "Textul introdus : "
30 PRINT T$
40 PRINT "Textul transpus : "
50 FOR I=1 TO LEN(T$)
60 C$=MID$(T$,I,1)
70 IF ASC(C$)>96 AND ASC(C$)<123 THEN GOTO 100
80 PRINT C$;
90 GOTO 110
100 PRINT CHR$(ASC(C$)-32);
110 NEXT I
120 END
```

2.10. Se dă un număr natural de maximum opt cifre. Să se construiască un şir de caractere care să exprime în litere numărul considerat.

Realizarea acestui program, fără a fi dificilă, necesită multă atenţie şi migală. Vom parcurge împreună paşii mai importanţi ai programului :

Se şterge ecranul, se iniţializează zona de date, se definesc expresiile necesare pentru exprimarea în litere a unui număr ; se poate vedea că, dacă lipseşte comanda CLEAR, la o eroare de introducere a datelor, se va obţine o eroare de dublare a definirii dimensiunilor tablourilor

care intervin ; la calculatoarele din familia HC, toate instrucțiunile GOTO 10 trebuie înlocuite cu RUN, pentru a evita această dublare :

```
10 CLS : CLEAR : DATA " zero"," unu"," doi"," trei"," patru","
cinci"," sase"," sapte"," opt"," noua"
20 DATA "","un","doi","trei","pai","cinci","sai","sapte","opt",
"noua"
30 DATA " o"," doua"," trei"," patru"," cinci"," sai"," sapte","
opt"," noua"
40 DATA " miliarde"," milioane"," mii"
50 DATA " miliard"," milion"," mie"
60 FOR I=1 TO 10 : READ H$(I) : NEXT I
70 FOR I=1 TO 10 : READ L$(I) : NEXT I
80 FOR I=1 TO 9 : READ M$(I) : NEXT I
90 FOR I=1 TO 3 : READ N$(I) : NEXT I
100 FOR I=1 TO 3 : READ P$(I) : NEXT I
```

Am ales introducerea numărului sub forma unui șir de caractere, pentru a putea introduce numere pînă la 12 cifre ; dacă dorim să introducem chiar numere, linia 110 poate fi înlocuită cu :

```
110 CLS ; INPUT "Introduceti numarul :"; A1: A$=LEFT$
(STR$(A1), LEN(STR$) - 1)
```

De asemenea, secvența validează și introducerea corectă a șirului de caractere care reprezintă numărul :

```
110 CLS : INPUT "Introduceti numarul (maxim 12 cifre) : "
A$
120 E=0 : IF LEN(A$)=0 OR LEN(A$)>12 THEN GOTO 110
130 FOR I=1 TO LEN(A$)
```

```
140 IF ASC(MID$(A$,I,1))<48 OR ASC(MID$(A$,I,1))>57
THEN E=1
150 NEXT I
```

Șirul de caractere care reprezintă numărul este împărțit în grupe de câte trei caractere care reprezintă miliarde, milioane, mii, unități :

```
160 IF E=1 THEN GOTO 110
170 C$="" : S=0
180 K=INT((LEN(A$) - 1)/3)
190 U$=RIGHT$(A$,3)
200 IF K=0 THEN GOTO 260
210 A$=LEFT$(A$,LEN(A$)-3) : M3$=RIGHT$(A$,3)
220 IF K=1 THEN GOTO 260
230 A$=LEFT$(A$,LEN(A$)-3) : M2$=RIGHT$(A$,3)
240 IF K=2 THEN GOTO 260
250 A$=LEFT$(A$,LEN(A$)-3) : M1$=RIGHT$(A$,3)
```

Se transformă, cu ajutorul subrutinei 400 - 510, fiecare grupă într-o secvență de caractere care exprimă în litere grupa respectivă :

```
260 X$=U$ : GOSUB 400 : V$=Y$
270 X$=M1$ : GOSUB 400 : N1$=Y$
280 X$=M2$ : GOSUB 400 : N2$=Y$
290 X$=M3$ : GOSUB 400 : N3$=Y$
```

Se face legătura gramaticală între grupele determinate anterior, se afișează expresia în litere, după care programul principal se termină :

```
300 R1$=N$(1) : IF VAL(M1$)=1 THEN R1$="un "+P$(1)
310 R2$=N$(2) : IF VAL(M2$)=1 THEN R2$="un "+P$(2)
320 R3$=N$(3) : IF VAL(M3$)=1 THEN R3$="o "+P$(3)
```

```
330 R1$=N1$+R1$ : R2$=N2$+R2$ : R3$=N3$+R3$
340 IF K<3 THEN R1$=""
350 IF K<2 THEN R2$=""
360 IF K<1 THEN R3$=""
370 PRINT R1$+R2$+R3$+V$
380 END
```

Subrutina 400 - 510 face transformarea fiecărei grupe în litere, această exprimare fiind dată de Y$; se analizează pe rînd cazurile :

- valoarea lui X$ reprezintă un număr natural mai mic decît 10;

- numărul determinat de ultimele două cifre din X$ este 10 ;

- numărul determinat de ultimele două cifre din X$ este cuprins între 10 și 20 ;

- numărul determinat de ultimele două cifre din X$ este mai mare ca 20 ;

Este analizată situația dată de cifra sutelor pentru a vedea dacă folosim singularul sau pluralul :

```
400 IF VAL(X$)<10 THEN Z$=H$(VAL(X$)+1) : Y$="" :
W$="" : GOTO 500
410 T$=RIGHT$(X$,2) : IF VAL(RIGHT$(T$,2))=10 THEN
Z$= "zece" : GOTO 460
420 IF VAL(RIGHT$(T$,2))<20 AND VAL(RIGHT$(X$,2))>
10 THEN Z$=L$(VAL(MID$(T$,2,1))+1)+"sprezece" :
GOTO 460
430 IF VAL(RIGHT$(T$,1))=0 THEN W$="" : GOTO 450
440 W$=H$(VAL(RIGHT$(T$,1))+1)
450 Z$=M$(VAL(MID$(T$,1,1)))+" zeci si"+W$ : IF VAL(T$)
=0 THEN W$=""
460 IF VAL(X$)<100 THEN Y$="" : W$="" : GOTO 500
470 IF VAL(X$)<200 THEN W$=" suta " : GOTO 490
480 W$=" sute "
```

```
490 Y$=M$(VAL(LEFT$(X$,1)))
500 Y$=Y$+W$+Z$ : IF VAL(X$)<2 THEN Y$=""
510 RETURN
```

2.11. Se consideră un număr natural de maxim patru cifre. Să se construiască un şir de caractere care să reprezinte numărul considerat în cifre romane.

Nici acest program nu este dificil, dar necesită şi el mai multă atenţie datorită multitudinii cazurilor care trebuie tratate. Lăsăm cititorului sarcina de a analiza programele 2.11 şi 2.12.

```
10 CLS : CLEAR : INPUT "Introduceti numarul : ",N
20 IF N<1 OR N>9999 OR N<>INT(N) THEN 10
30 A(1)=N - INT(N/10)*10 : N=INT(N/10)
40 A(2)=N - INT(N/10)*10 : N=INT(N/10)
50 A(3)=N - INT(N/10)*10 : N=INT(N/10)
60 A(4)=N - INT(N/10)*10 : N=INT(N/10)
70 N$=""
80 IF A(4)<>0 THEN 120
90 IF A(3)<>0 THEN 130
100 IF A(2)<>0 THEN 180
110 K=1 : GOTO 230
120 FOR I=1 TO A(4) : N$=N$+"M" : NEXT I
130 IF A(3)=9 THEN N$=N$+"CM" : GOTO 180
140 IF A(3)=4 THEN N$=N$+"CD" : GOTO 180
150 IF A(3)>=5 THEN N$=N$+"D"
160 A(3)=A(3)-5*INT(A(3)/5) : IF A(3)=0 THEN 180
170 FOR I=1 TO A(3) : N$=N$+"C" : NEXT I
180 IF A(2)=9 THEN N$=N$+"XC" : GOTO 230
190 IF A(2)=4 THEN N$=N$+"XL" : GOTO 230
200 IF A(2)>=5 THEN N$=N$+"L"
```

```
210 A(2)=A(2)-5*INT(A(2)/5) : IF A(2)=0 THEN 230
220 FOR I=1 TO A(2) : N$=N$+"X" : NEXT I
230 IF A(1)=9 THEN N$=N$+"IX" : GOTO 280
240 IF A(1)=4 THEN N$=N$+"IV" : GOTO 280
250 IF A(1)>=5 THEN N$=N$+"V"
260 A(1)=A(1)-5*INT(A(1)/5) : IF A(1)=0 THEN 280
270 FOR I=1 TO A(1) : N$=N$+"I" : NEXT I
280 PRINT "Numarul scris cu cifre romane este : ";N$
```

2.12. Se consideră un şir de caractere care reprezintă un număr scris cu cifre romane. Să se exprime numărul în scrierea arabă.

```
10 CLS : CLEAR ; INPUT N$
20 IF LEN(N$)=0 THEN 10
30 A$="MDCLXVI"
40 E=0 : FOR I=1 TO LEN(N$)
50 IF INSTR(A$,MID$(N$,I,1))=0 THEN E=1
60 NEXT I
70 IF E=1 THEN GOTO 10
80 N=0
90 I=1
100 P=0
110 IF MID$(N$,I,1)<>"M" THEN 130
120 P=P+1 : N=N+1000 : I=I+1 : IF I<=LEN(N$) THEN
110
130 IF P>9 THEN 10
140 IF I=LEN(N$) THEN 200
150 IF I>LEN(N$) THEN 470
160 IF MID$(N$,I+1,1)="M" AND MID$(N$,I,1)<>"C" THEN
10
```

```
170 IF MID$(N$,I+1,1)="M" AND MID$(N$,I,1)="C" THEN
N= N+900 : I=I+2 : GOTO 250
180 IF MID$(N$,I+1,1)="D" AND MID$(N$,I,1)<>"C" THEN
10
190 IF MID$(N$,I+1,1)="D" AND MID$(N$,I,1)="C" THEN
N= N+400 : I=I+2 : GOTO 250
200 IF MID$(N$,I,1)="D" THEN N=N+500 : I=I+1
210 P=0
220 IF MID$(N$,I,1)<>"C" THEN 240
230 P=P+1 : N=N+100 : I=I+1 : IF I<=LEN(N$) THEN
220
240 IF P>3 THEN 10
250 IF I=LEN(N$) THEN 310
260 IF I>LEN(N$) THEN 470
270 IF MID$(N$,I+1,1)="C" AND MID$(N$,I,1)<>"X" THEN
10
280 IF MID$(N$,I+1,1)="C" AND MID$(N$,I,1)="X" THEN
N= N+90 : I=I+2 : GOTO 350
290 IF MID$(N$,I+1,1)="L" AND MID$(N$,I,1)<>"X" THEN
10
300 IF MID$(N$,I+1,1)="L" AND MID$(N$,I,1)="X" THEN
N= N+40 : I=I+2 : GOTO 350
310 IF MID$(N$,I,1)="L" THEN N=N+50 : I=I+1
320 P=0
330 IF MID$(N$,I,1)<>"X" THEN 350
340 P=P+1 : N=N+10 : I=I+1 : IF I<=LEN(N$) THEN 330
350 IF P>3 THEN 10
360 IF I=LEN(N$) THEN 430
370 IF I>LEN(N$) THEN 470
```

```
380 IF MID$(N$,I+1,1)="X" AND MID$(N$,I,1)<>"I" THEN
10
390 IF MID$(N$,I+1,1)="X" AND MID$(N$,I,1)="I" THEN
N= N+9 : I=I+2 : GOTO 470
400 IF MID$(N$,I+1,1)="V" AND MID$(N$,I,1)<>"I" THEN
10
410 IF MID$(N$,I+1,1)="V" AND MID$(N$,I,1)="I" THEN
N= N+4 : I=I+2 : GOTO 470
420 IF MID$(N$,I;1)="V" THEN N=N+5 : I=I+1
430 P=0
440 IF MID$(N$,I,1)<>"I" THEN 460
450 P=P+1 : N=N+1 : I=I+1 : IF I<=LEN(N$) THEN 440
460 IF P>3 THEN 10
470 IF I<=LEN(N$) THEN 10
480 PRINT N
```

2.3. PROBLEME DE DIVIZIBILITATE.

3.1. Se consideră un număr natural *n*. Să se facă un program BASIC pentru scrierea lui într-o bază de numerație *b* dată.

Programul, deși scurt, prezintă o idee deosebită care evită tratarea separată a cazurilor în care baza este mai mică decît 10 și cînd este mai mare decît 10 : tratarea cifrelor drept caractere. Este tratat numai cazul în care baza de numerație nu depășește 16, dar programul poate fi extins cu ușurință și la baze mai mari.

Se șterge ecranul, se inițializează zona de date, se introduce numărul, se introduce baza de numerație și se validează datele introduse :

```
10 CLS : CLEAR : INPUT "Introduceti numarul :" ; N : INPUT
"Introduceti baza de numeratie :" ; B
20 IF B<2 OR B>16 OR N<0 OR INT(N)<>N OR INT(B)
<>B THEN GOTO 10
```

Se definește o variabilă care conține cifrele posibile ale bazei de numerație în ordine crescătoare și se definesc cifrele în baza de numerație respectivă :

```
30 LET B$ = "0123456789ABCDEF" : LET B$ = LEFT$(B$,
B)
```

Se inițializează o variabilă de tip șir de caractere care va reprezenta numărul în baza de numerație indicată cu șirul vid și se copie valoarea lui N într-o nouă variabilă numerică :

```
40 A$ = ""
50 LET A=N
```

Ultima cifră, în baza de numerație indicată, este tocmai restul împărțirii lui A la baza de numerație B ; se construiește variabila A$

prin concatenarea cifrei obținute la vechiul A$; se determină cîtul împărțirii lui A la B și se păstrează în A ; procedeul continuă pînă cînd A<B, adică A este cifră în baza de numerație indicată :

```
60 IF A<B THEN GOTO 80
70 LET R=A - INT(A/B)*B : LET A$=MID$(B$,R+1,1)+A$
: LET A=INT(A/B) : GOTO 60
```

Se concatenează la A$ ultima cifră :

```
80 LET A$=MID$(B$,A+1,1)+A$
```

Se afișează A$:

```
90 PRINT A$
```

3.2. Să se facă un program BASIC care să efectueze cele patru operații cu fracții. Rezultatul va fi scris sub formă de fracție ireductibilă. Operațiile vor fi efectuate utilizînd numai algoritmii învățați în clasa a V-a.

Programul nu este dificil, el constituind un exercițiu pentru cititor dacă a analizat pînă acum programele prezentate și cunoaște algoritmul lui EUCLID.

Se introduc cele două fracții ca șiruri de caractere, fiecare fiind format din trei subșiruri concatenate, primul reprezintă un număr natural care este numărătorul, al doilea este "/", iar al treilea reprezintă numitorul ; la validarea fiecărui șir se va avea grijă să se verifice următoarele :

- *codurile ASCII ale caracterelor trebuie să fie cuprinse între 47 și 57 (cifrele de la 0 la 9 au codurile ASCII de la 48 la 57, iar "/" are codul 47) ;*
- *caracterul "/" trebuie să apară exact o dată ;*
- *numitorul să fie diferit de 0 :*

```
10 CLS : CLEAR
20 INPUT "Introduceti prima fractie (p/q)"; A$
30 IF LEN(A$)<3 THEN GOTO 10
40 E=0 : FOR I=1 TO LEN(A$)
50 IF ASC(MID$(A$,I,1))>57 OR ASC(MID$(A$,I,1))<47
THEN LET E=1
60 NEXT I
70 IF E=1 THEN GOTO 10
80 K=0
90 FOR I=1 TO LEN(A$)
100 IF MID$(A$,I,1)<>"/" THEN GOTO 120
110 K=K+1 : A=I
120 NEXT I
130 IF K<>1 THEN GOTO 10
140 A1=VAL(LEFT$(A$,A - 1)) : A2=VAL(RIGHT$(A$,
LEN(A$) - A))
150 IF A2=0 THEN GOTO 10
160 INPUT "Introduceti a doua fractie (p/q)";B$
170 IF LEN(B$) < 3 THEN GOTO 10
180 E=0 : FOR I=1 TO LEN(B$)
190 IF ASC(MID$(B$,I,1))>57 OR ASC(MID$(B$,I,1))<47
THEN LET E=1
200 NEXT I
210 IF E=1 THEN GOTO 10
220 K=0
230 FOR I=1 TO LEN(B$)
240 IF MID$(B$,I,1)<>"/" THEN GOTO 260
250 K=K+1 : B=I
260 NEXT I
270 IF K<>1 THEN GOTO 10
```

Se determină valoarea efectivă a numărătorilor și numitorilor celor două fracții :

```
280 B1=VAL(LEFT$(B$,B - 1)) : B2=VAL(RIGHT$(B$,
LEN(B$) - B))
290 IF B2=0 THEN GOTO 10
300 INPUT "Introduceti operatia (+,-,*,:)"; C$
310 IF LEN(C$)<>1 THEN GOTO 10
```

Se introduce și se validează operația care trebuie efectuată:

```
320 S$="+-*:" : E=0 : FOR I=1 TO 4
330 IF MID$(S$,I,1)=C$ THEN E=1
340 NEXT I
350 IF E=0 THEN GOTO 10
```

Se cercetează ca în cazul împărțirii, valoarea celei de-a doua fracții să nu fie 0:

```
360 IF C$=":" AND B1=0 THEN GOTO 10
```

Se selectează operația, făcîndu-se trimiterea la ramura respectivă:

```
370 IF C$="+" THEN GOTO 410
380 IF C$="-" THEN GOTO 420
390 IF C$="*" THEN GOTO 430
400 IF C$=":" THEN GOTO 440
```

Se determină numărătorul și numitorul fracției obținute ca rezultat și se apelează subrutina 450 - 530 care face simplificarea și afișarea rezultatului :

```
410 M=A1*B2+A2*B1 : N=A2*B2 : GOSUB 450 : END
```

```
420 M=A1*B2-A2*B1 : N=A2*B2 : GOSUB 450 : END
430 M=A1*B1 : N=A2*B2 : GOSUB 450 : END
440 M=A1*B2 : N=A2*B1 : GOSUB 450 : END
```

Subrutina este formată din trei părți :

- Se aplică algoritmul lui EUCLID pentru determinarea c.m.m.d.c. dintre numărător și numitor :

```
450 U=M : V=N : IF M<N THEN GOTO 470
460 L=M : M=N : N=L
470 R=N - M*INT(N/M)
480 IF R=0 THEN GOTO 500
490 N=M : M=R : GOTO 470
```

Se simplifică fracția prin c.m.m.d.c.determinat anterior :

```
500 U=U/M : V=V/M
```

Se construiește și se afișează un șir care reprezintă fracția rezultat:

```
510 D$=STR$(U)+"/"+STR$(V)
520 PRINT A$+C$+B$+"="+D$
530 RETURN
```

3.3. Să se facă un program pentru calculul sumei numerelor reprezentate de cifrele unui număr dat. Nu este permisă introducerea sau utilizarea șirurilor de caractere pentru a defini numere.

Programul se bazează pe faptul că ultima cifră a unui număr este restul împărțirii acestui număr la 10. Algoritmul este evident :

- se determină ultima cifră a numărului , care se adună la S care a fost inițializată cu 0 ;

- se calculează cîtul împărțirii numărului la 10, obținîndu-se un

număr care conține, în aceeași ordine, cifrele numărului precedent, mai puțin ultima cifră ;

- *se repetă procedeul pînă cînd se obține 0 ;*
- *se afișează S :*

```
10 INPUT "Introduceti numarul :"; N
20 LET S=0
30 IF N=0 THEN PRINT "Suma este : "S : END
40 LET S=S+(N - INT(N/10)*10)
50 LET N=INT(N/10)
60 IF N=0 THEN GOTO 80
70 GOTO 40
80 PRINT "Suma este : "; S
```

Problemele 3.4 și 3.5. sînt o aplicație directă a problemei 3.3.

3.4. Utilizînd numai criteriile de divizibilitate din clasa a V-a, să se stabilească dacă un număr natural *n* se divide prin 2, 3, 4, 5, 9, 25.

```
10 CLS
20 INPUT "Introduceti numarul :"; N
30 IF N=0 THEN PRINT "Numarul se divide la 2,3,4,5,9,25"
: END
40 LET Q=N - INT(N/10)*10
50 IF Q=INT(Q/2)*2 THEN PRINT "Numarul se divide la 2
" : GOTO 70
60 PRINT "Numarul nu se divide la 2 "
70 IF Q=INT(Q/5)*5 THEN PRINT "Numarul se divide la 5
" : GOTO 90
80 PRINT "Numarul nu se divide la 5 "
90 LET P=N - INT(N/100)*100
```

```
100 IF P=INT(P/25)*25 THEN PRINT "Numarul se divide la
25 " : GOTO 120
110 PRINT "Numarul nu se divide la 25 "
120 IF P=INT(P/4)*4 THEN PRINT "Numarul se divide la 4
" : GOTO 140
130 PRINT "Numarul nu se divide la 4 "
140 LET S=0
150 LET S=S+(N - INT(N/10)*10)
160 LET N=INT(N/10)
170 IF N=0 THEN GOTO 190
180 GOTO 150
190 IF S=INT(S/3)*3 THEN PRINT "Numarul se divide la 3
" : GOTO 210
200 PRINT "Numarul nu se divide la 3 "
210 IF S=INT(S/9)*9 THEN PRINT "Numarul se divide la 9
" : GOTO 230
220 PRINT "Numarul nu se divide la 9 "
230 END
```

3.5. Se cunoaște următorul criteriu de divizibilitate prin 11 a unui număr natural : *Un număr natural n se divide prin 11 ⇔ suma numerelor reprezentate de cifrele de pe locurile pare, din care se scade suma numerelor reprezentate de cifrele de pe locurile impare, este un număr care se divide prin 11.* Să se facă un program, care să utilizeze acest criteriu pentru a stabili dacă un număr natural dat se divide sau nu la 11.

```
10 INPUT "Introduceti numarul : "; N
20 IF N=0 THEN PRINT "Numarul se divide la 11 " : END
30 LET S=0
40 LET P=0
```

```
50 LET S=S+(N - INT(N/10)*10)
60 LET N=INT(N/10)
70 IF N=0 THEN GOTO 120
80 LET P=P+(N - INT(N/10)*10)
90 LET N=INT(N/10)
100 IF N=0 THEN GOTO 120
110 GOTO 50
120 IF (S - P) - INT((S - P)/11)*11 =0 THEN PRINT "Numarul
se divide la 11 " : END
130 PRINT "Numarul nu se divide la 11 "
140 END
```

3.6. Să se facă un program cu ajutorul căruia să se efectueze cele patru operații cu numere mari (oricîte cifre). În cazul împărțirii se va specifica cîtul și restul.

Problema în sine este mai dificilă decît celelalte. Dacă s-ar pune numai problema scăderii, totul ar fi foarte simplu, dar în privința celorlalte operații, lucrurile se complică, mai ales la împărțire, unde se cere cîtul și restul. Timpul de lucru pentru o operație crește o dată cu numărul de cifre ale numerelor introduse. Vom urmări împreună acest program.

Se șterge ecranul, se inițializează zona de variabile, se introduce primul număr sub formă de șir de caractere A$, se validează introducerea acestui număr, se introduce al doilea număr sub forma șirului de caractere B$ și se validează introducerea acestui număr :

- după inițializarea calculatorului și după introducerea primului șir A$ se verifică dacă acesta nu este șirul vid (linia 30) :

```
10 CLS : CLEAR
20 INPUT "Introduceti primul numar : "; A$
30 IF LEN(A$)=0 THEN GOTO 10
```

- se verifică dacă caracterele lui A$ sînt cifre (dacă au codul ASCII cuprins între 47 şi 58, exclusiv) :

```
40 E=0 : FOR I=1 TO LEN(A$)
50 IF ASC(MID$(A$,I,1))<48 OR ASC(MID$(A$,I,1))>57
THEN E=1
60 NEXT I
70 IF E=1 THEN GOTO 10
```

- după introducerea celui de-al doilea şir B$ se verifică dacă acesta nu este şirul vid (linia 90) :

```
80 INPUT "Introduceti al doilea numar : "; B$
90 IF LEN(B$)=0 THEN GOTO 10
```

- se verifică dacă caracterele lui B$ sînt cifre (dacă au codul ASCII cuprins între 47 şi 58, exclusiv) :

```
100 E=0 : FOR I=1 TO LEN(B$)
110 IF ASC(MID$(B$,I,1))<48 OR ASC(MID$(B$,I,1))>57
THEN E=1
120 NEXT I
130 IF E=1 THEN GOTO 10
```

Se determină numărul maxim de cifre ale celor două numere, se dimensionează patru tablouri unidimensionale pentru cifrele celor două numere şi pentru rezultate, se determină cifrele primului număr în A şi cifrele celui de-al doilea număr în B :

- se ia pentru N valoarea maximă dintre lungimile şirurilor A$ şi B$ şi se dimensionează cinci variabile de tip tablou unidimensional (A care va păstra cifrele lui A$, B care va păstra cifrele lui B$, D, E şi F necesare unor operaţii intermediare) :

```
140 N=LEN(A$) : IF LEN(B$)>LEN(A$) THEN N=LEN(B$)
```

```
150 DIM A(N) : DIM B(N) : DIM D(2*N+1) : DIM E(2*N+1)
: DIM F(2*N+1)
```

- se determină cifrele numerelor reprezentate de A$ (liniile 160 - 180) şi de B$ (liniile 190 - 210) :

```
160 FOR I=1 TO LEN(A$)
170 A(I)=VAL(MID$(A$,LEN(A$) - I+1,1))
180 NEXT I
190 FOR I=1 TO LEN(B$)
200 B(I)=VAL(MID$(B$,LEN(B$) - I+1,1))
210 NEXT I
```

Se introduce simbolul operaţiei şi se validează introducerea acestuia :

```
230 INPUT "Introduceti operatia(+, - ,*,:) : "; C$
240 IF LEN(C$)<>1 THEN GOTO 10
250 S$="+ - *:" : E=0 : FOR I=1 TO 4
260 IF MID$(S$,I,1)=C$ THEN  LET E=1
270 NEXT I
280 IF E=0 THEN GOTO 10
```

Se selectează operaţia, programul ramificîndu-se :
- pentru adunare se execută ramura care începe la linia 330 :

```
290 IF C$="+" THEN GOTO 330
```

- pentru diferenţă se execută subrutina 480 - 560 şi apoi se afişează rezultatul :

```
300 IF C$=" - " THEN GOSUB 480 : PRINT "Diferenta este
:"; D$ : END
```

- înmulțirea este executată de partea din program 570 - 970 :

```
310 IF C$="*" THEN GOTO 570
```

- împărțirea este efectuată de ramura de program 1190-1550 :

```
320 GOTO 1190
```

Prezentăm, în continuare, cele patru ramuri ale programului :

Prima ramură reprezintă adunarea ; ținînd cont că cifra de pe locul I a sumei este restul împărțirii sumei cifrelor celor două numere de pe același loc, adunate cu restul anterior, la 10, iar noul rest este cîtul ; se inițializează mai întîi restul R cu 0 și se aplică acest algoritm :

```
330 LET R=0 : GOSUB 340 : GOTO 460
```

- subrutina 340 - 450 efectuează efectiv suma, determinîndu-se efectiv cifrele acesteia :

```
340 FOR I=1 TO N
350 LET D(I)=A(I)+B(I)+R : LET R=INT(D(I)/10) : LET D(I)=
D(I) - INT(D(I)/10)*10
360 NEXT I
370 LET D(N+1)=R
```

- se definește o nouă variabilă de tip șir de caractere D$ care constituie rezultatul adunării și care se afișează ; deoarece funcția STR$ introduce un spațiu suplimentar în stînga, trebuie eliminat acest spațiu ; de asemenea, sînt eliminate zerourile nesemnificative (cele din stînga numărului, dacă acestea există) ; secvența a fost organizată ca o subrutină, deoarece adunarea este necesară la efectuarea celorlalte operații :

```
380 LET D$=""
390 FOR I=1 TO N+1
400 LET D$=RIGHT$(STR$(D(I)),1)+D$
410 NEXT I
420 IF LEFT$(D$,1)<>"0" AND LEN(D$)>1 THEN GOTO
450
430 IF LEN(D$)=1 THEN GOTO 450
440 D$=RIGHT$(D$,LEN(D$) - 1) : GOTO 420
450 RETURN
```

- se afișează rezultatul adunării :

```
460 PRINT "Suma este : ";D$ : END
470 RETURN
```

Pentru diferență, a doua ramură, se stabilește mai întîi care din cele două numere este mai mare pentru a stabili semnul rezultatului ; Pentru o execuție mai rapidă, operația de scădere este transformată în adunare în modul următor : presupunînd că numărul reprezentat de A$ este mai mare decît cel reprezentat de B$, avem :

$$x - y = x + (999 \ldots 9 - y) + 1 - 1000 \ldots 0$$

Considerînd restul inițial R ca fiind 1, al doilea număr ca fiind complementul față de 9 (adică cifrele complementului se obțin prin scăderea cifrelor numărului considerat din 9) și, neglijînd din rezultat, prima cifră care va fi 1, scăderea a fost transformată în adunare:

- se stabilește care din numerele reprezentate de A$ și B$ este mai mare ; dacă B$ este mai mare, se atribuie lui U$ (semnul diferenței) semnul " - "; în caz contrar se atribuie lui U$ valoarea "", deci fără semn:

```
480 IF LEN(A$)<LEN(B$) THEN U$=" - " : GOTO 510
490 IF LEN(A$)>LEN(B$) THEN U$="" : GOTO 510
500 U$="" : IF A$<B$ THEN U$=" - "
```

- se calculează complementul față de 9 a celui mai mic dintre numerele reprezentate de A$ și B$:

```
510 R=1
520 IF U$=" - " THEN GOTO 540
530 FOR I=1 TO N : B(I)=9 - B(I) : NEXT I : GOTO 550
540 FOR I=1 TO N : A(I)=9 - A(I) : NEXT I
```

- se calculează suma dintre cel mai mare număr reprezentat de A$ și complementul față de 9 a celui mai mic, obținîndu-se rezulta- tul în variabila D$; deoarece rezultatul va conține un 1 în plus și, după scoaterea acestui 1 (practic se scade 1000 . . . 0), se adaugă semnul reprezentat de U$:

```
550 GOSUB 340 : D$=U$+RIGHT$(D$,LEN(D$) - 1) :
GOSUB 420
560 RETURN
```

Ramura următoare realizează înmulțirea. Înmulțirea este, în mod evident o adunare repetată, dar aceasta ar conduce la un număr prea mare de operații ; dar se poate aplica algoritmul obișnuit prin înmulți- rea unui număr cu cifrele celuilalt, sumele obținute fiind multiplicate cu 100...0, numărul de 0 fiind dat de poziția cifrei respective ; sumele parțiale, astfel obținute, se adună obținîndu-se rezultatul cerut :

- se determină care dintre numerele reprezentate de A$ și B$ are mai puține cifre ; se atribuie lui M lungimea numărului cu cele mai puține cifre ; dacă A$ are lungimea mai mică, este bine, în caz contrar se schimbă A$ cu B$:

```
570 M=LEN(A$) : IF LEN(A$)>LEN(B$) THEN M=LEN(B$)
: E$=A$ : A$=B$ : B$=E$
```

- se determină cifrele lui A$ și ale lui B$:

```
580 FOR I=1 TO LEN(A$)
```

```
590 A(I)=VAL(MID$(A$,LEN(A$) - I+1,1))
600 NEXT I
610 FOR I=1 TO LEN(B$)
620 B(I)=VAL(MID$(B$,LEN(B$) - I+1,1))
630 NEXT I
```

- se inițializează rezultatul E$ cu "0", R care reprezintă transportul la rangul superior cu 0 și S$ cu "000... 0" care reprezintă numărul de zerouri ce se vor adăuga la produsele parțiale înainte de a fi adunate la rezultatul final, S$:

```
640 E$="0" : S$=""
650 FOR J=1 TO M
660 S$=S$+"0" : R=0
```

- se calculează o sumă parțială :

```
670 FOR I=1 TO N
680 D(I)=A(J)*B(I)+R : R=INT(D(I)/10) : D(I)=D(I) - 10*R
690 NEXT I
700 IF R=0 THEN H$="" : GOTO 720
710 H$=RIGHT$(STR$(R),1)
```

- se determină o sumă parțială D$:

```
720 LET D$=""
730 FOR I=1 TO N
740 LET D$=RIGHT$(STR$(D(I)),1)+D$
750 NEXT I
760 D$=H$+D$+S$ : D$=LEFT$(D$,LEN(D$) - 1)
```

- se determină cifrele lui D$:

```
770 FOR I=1 TO LEN(D$)
780 D(I)=VAL(MID$(D$,LEN(D$) - I+1,1))
790 NEXT I
```

- se calculează cifrele rezultatului anterior :

```
800 FOR I=1 TO LEN(E$)
810 E(I)=VAL(MID$(E$,LEN(E$) - I+1,1))
820 NEXT I
```

- se calculează suma dintre vechiul rezultat şi produsul parţial :

```
830 T=LEN(D$) : IF LEN(E$)>LEN(D$) THEN T=LEN(E$)
840 W=0
850 FOR I=1 TO T
860 LET F(I)=D(I)+E(I)+W : LET W=INT(F(I)/10) : LET
F(I)= F(I) - INT(F(I)/10)*10
870 NEXT I
```

- se determină valoarea finală pentru E$ care se afişează :

```
880 LET F(T+1)=W
890 LET E$=""
900 FOR I=1 TO T+1
910 LET E$=RIGHT$(STR$(F(I)),1)+E$
920 NEXT I
930 IF LEFT$(E$,1)<>"0" AND LEN(E$)>1 THEN GOTO
960
940 IF LEN(E$)=1 THEN GOTO 960
950 E$=RIGHT$(E$, LEN(E$) - 1) : GOTO 930
```

```
960 NEXT J
970 PRINT E$ : END
```

Ultima ramură este rezervată împărțirii. Pentru această operație am ales algoritmul învățat în clase primare; lăsăm cititorul ca, amintindu-și cum se face împărțirea, să reconstituie acest algoritm după secvența de program prezentată mai jos:

```
1190 X$=A$ : Y$=B$ : Q$=""
1200 M$=X$ : N$=Y$
1210 Q=0
1220 IF LEN(M$)<LEN(N$) THEN 1380
1230 IF LEN(M$)>LEN(N$) THEN 1290
1240 IF M$<N$ THEN 1380
1250 K1=LEN(B$)
1260 A$=M$ : FOR I=1 TO N : A(I)=0 : NEXT I : FOR I=1
TO LEN(A$) : A(I)=VAL(MID$(A$,LEN(A$)+1,1)) : NEXT I
1270 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1
TO LEN(B$) : B(I)=VAL(MID$(B$,LEN(B$)+1,1)) : NEXT I
1280 GOTO 1430
1290 K1=LEN(B$)
1300 A$=LEFT$(M$,LEN(B$)) : FOR I=1 TO N : A(I)=0 :
NEXT I : FOR I=1 TO LEN(A$) : A(I)=VAL(MID$(A$,
LEN(A$)+1,1)) : NEXT I
1310 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1
TO LEN(B$) : B(I)=VAL(MID$(B$,LEN(B$)+1,1)) : NEXT I
1320 IF A$<B$ THEN 1340
1330 GOTO 1430
1340 K1=LEN(B$)+1
1350 A$=LEFT$(M$,LEN(B$)+1) : FOR I=1 TO N : A(I)=0
```

```
: NEXT I : FOR I=1 TO LEN(A$) : A(I)=VAL(MID$(A$,
LEN(A$)+1, 1)) : NEXT I
1360 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1
TO LEN(B$) : B(I)=VAL(MID$(B$,LEN(B$)+1,1)) : NEXT I
1370 GOTO 1430
1380 LET D$=Q$
1390 IF LEFT$(D$,1)<>"0" AND LEN(D$)>1 THEN GOTO
1420
1400 IF LEN(D$)=1 THEN GOTO 1420
1410 D$=RIGHT$(D$,LEN(D$)-1) : GOTO 1150
1420 PRINT "Citul este : ";D$,"Restul este : ";A$ : END
1430 D$="0" : IF LEN(B$)=1 AND VAL(B$)=0 THEN GOTO
10
1440 M1$=B$
1450 Q=0
1460 IF LEN(A$)<LEN(B$) THEN GOTO 1540
1470 IF LEN(A$)>LEN(B$) THEN GOTO 1490
1480 IF A$<B$ THEN 1540
1490 Q=Q+1 : GOTO 1500
1500 GOSUB 480
1510 A$=D$ : FOR I=1 TO N : A(I)=0 : NEXT I : FOR I=1
TO LEN(A$) : A(I)=VAL(MID$(A$,LEN(A$)+1,1)) : NEXT I
1520 B$=M1$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1
TO LEN(B$) : B(I)=VAL(MID$(B$,LEN(B$)+1,1)) : NEXT I
1530 GOTO 1460
1540 Q$=Q$+RIGHT$(STR$(Q),1)
1550 M$=A$+RIGHT$(M$,LEN(M$)-K1) : N$=Y$ : GOTO
1210
```

3.7. Să se calculeze 1000 ! (prin *n* ! înţelegem produsul numerelor naturale de la 1 la *n*, unde *n* este un număr natural).

Pentru calculul lui 1000 ! am folosit calculul în baza de numeraţie 10000 ; trebuie avut grijă să nu neglijăm zerourile nesemnificative ale "cifrelor" :

```
10 CLS : CLEAR : DIM A(1000) : INPUT N
20 IF N<0 OR N<>INT(N) THEN GOTO 10
30 IF N=0 OR N=1 THEN PRINT N; " ! = ";1 : END
40 FOR I=2 TO 1000 : A(I)=0 : NEXT I
50 A(1)=1 : L=1
60 FOR K=1 TO N
70 R2=0 : R1=0 : I=1
80 IF R2<>0 THEN GOTO 100
90 IF I>L THEN GOTO 160
100 R=A(I)*K+R2
110 R2=INT(R/10000)
120 R1=R - R2*10000
130 A(I)=R1
140 I=I+1
150 GOTO 80
160 L=I - 1
170 NEXT K
180 PRINT N;" ! = "; : FOR J=1 TO I - 1
190 A$=STR$(A(I-J)) : A$="000000"+RIGHT$(A$,LEN(A$)
- 1) : A$=RIGHT$(A$,4)
200 IF J>1 THEN GOTO 250
210 FOR T=1 TO LEN(A$)
220 IF LEFT$(A$,1)<>"0" THEN GOTO 240
230 A$=RIGHT$(A$,LEN(A$) - 1)
```

```
240 NEXT T
250 PRINT A$;
260 NEXT J
```

De exemplu, pentru n=100, se obține :

100 ! = 93226215443944152681699238856266700490715968264381621485929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000000

3.8. Să se descompună în factori primi un număr natural *n* dat.

```
10 CLS : INPUT "Introduceti numarul :"; N
20 IF N<2 OR INT(N)<>N THEN GOTO 10
30 DIM C(N)
40 FOR I=2 TO N : C(I)=I :  NEXT I
50 FOR I=2 TO INT(N/2)
60 IF C(I)=0 THEN GOTO 100
70 FOR J=2*I TO N STEP I
80 C(J)=0
90 NEXT J
100 NEXT I
110 CLS
120 FOR I=2 TO N
130 IF C(I)=0 THEN GOTO 200
140 P=1 : K=0
150 P=P*C(I)
160 IF N/P<>INT(N/P) THEN GOTO 180
170 K=K+1 : GOTO 150
180 IF K=0 THEN  GOTO 200
```

```
190 PRINT "factorul prim : "; C(I),"ordin de multiplicitate :
"; K
200 NEXT I
```

3.9. Să se determine toate numerele prime mai mici decît un număr natural dat.

```
10 CLS : INPUT "Introduceti numarul : "; N
20 IF N<2 OR INT(N)<>N THEN GOTO 10
30 DIM C(N)
40 FOR I=2 TO N : C(I)=I : NEXT I
50 FOR I=2 TO INT(N/2)
60 IF C(I)=0 THEN GOTO 100
70 FOR J=2*I TO N STEP I
80 C(J)=0
90 NEXT J
100 NEXT I
110 CLS
120 PRINT "Numerele prime mai mici sau egale cu "; N; "
sint :" : PRINT
130 FOR I=2 TO N
140 IF C(I)=0 THEN GOTO 160
150 PRINT C(I),
160 NEXT I
```

3.10. Să se determine toate numerele naturale perfecte mai mici decît un număr natural dat (prin număr perfect înţelegem un număr natural în care suma tuturor divizorilor pozitivi ai numărului este egală cu dublul numărului considerat).

```
10 CLS : INPUT "Introduceti numarul : "; N
20 FOR K=2 TO N
```

```
30 S=0
40 FOR I=1 TO K/2
50 IF K/I=INT(K/I) THEN S=S+I
60 NEXT I
70 IF S<>K THEN GOTO 90
80 PRINT "Numarul "; K ; " este perfect"
90 NEXT K
```

3.11. Să se determine toţi divizorii unui număr natural dat.

```
10 CLS : INPUT N
20 IF N<1 OR N<>INT(N) THEN 10
30 PRINT "Divizorii naturali ai numarului "; N ; " sint :"
40 FOR I=1 TO N
50 IF N/I=INT(N/I) THEN PRINT I,
60 NEXT I
```

3.12. Să se găsească toate solutiile în numere naturale, mai mici decît 200 ale ecuaţiei :

$$5 \cdot x - 3 \cdot y = 1.$$

```
10 CLS : INPUT "a="; A : INPUT "b="; B : INPUT "c="; C
20 INPUT "limita maxima :"; L
30 IF A<1 OR A<>INT(A) THEN GOTO 10
40 IF B<1 OR B<>INT(B) THEN GOTO 10
50 IF C<1 OR C<>INT(C) THEN GOTO 10
60 IF L<1 OR L<>INT(L) THEN GOTO 10
70 PRINT"Ecuatia : "; A ; "*x-"; B ; "*y="; C ; "  are
urmatoarele solutii in numere naturale <= cu "; L
80 FOR Y=1 TO L
90 X=(C+B*Y)/A
100 IF X<>INT(X) THEN GOTO 120
```

```
110 PRINT"x="; X, "y="; Y
120 NEXT Y
```

4. PROBLEME DIVERSE.

Pentru problemele din această secțiune vom prezenta numai programele ce vor putea fi descifrate acum mai ușor, după ce ați parcurs celelalte secțiuni ale cărții.

4.1. Să se facă un program prin care să se efectueze cele patru operații cu numere naturale scrise în baza de numerație 7, fără a trece numerele în baza 10. La împărțire se va indica câtul și restul.

```
10 CLS : CLEAR
20 INPUT "Introduceti primul numar : "; A$
30 IF LEN(A$)=0 THEN GOTO 10
40 E=0 : FOR I=1 TO LEN(A$)
50 IF ASC(MID$(A$,I,1))<48 OR ASC(MID$(A$,I,1))>54 THEN E=1
60 NEXT I
70 IF E=1 THEN GOTO 10
80 INPUT "Introduceti al doilea numar : "; B$
90 IF LEN(B$)=0 THEN GOTO 10
100 E=0 : FOR I=1 TO LEN(B$)
110 IF ASC(MID$(B$,I,1))<48 OR ASC(MID$(B$,I,1))>54 THEN E=1
120 NEXT I
130 IF E=1 THEN GOTO 10
140 N=LEN(A$) : IF LEN(B$)>LEN(A$) THEN N=LEN(B$)
150 DIM A(N) : DIM B(N) : DIM F(2*N+1) : DIM D(2*N+1) : DIM E(2*N+1)
160 FOR I=1 TO LEN(A$)
170 A(I)=VAL(MID$(A$,LEN(A$) - I+1,1))
180 NEXT I
190 FOR I=1 TO LEN(B$)
200 B(I)=VAL(MID$(B$,LEN(B$) - I+1,1))
210 NEXT I
220 IF A(LEN(A$))=0 OR B(LEN(B$))=0 THEN GOTO 10
230 INPUT "Introduceti operatia(+, - ,*,:) : ";C$
240 IF LEN(C$)<>1 THEN GOTO 10
```

```
250 S$="+ - *:" : E=0 : FOR I=1 TO 4
260 IF MID$(S$,I,1)=C$ THEN  LET E=1
270 NEXT I
280 IF E=0 THEN GOTO 10
290 IF C$="+" THEN GOTO 330
300 IF C$=" - " THEN GOSUB 480 : PRINT"Diferenta este  : "; D$ : END
310 IF C$="*" THEN GOTO 570
320 GOTO 980
330 LET R=0 : GOSUB 340 : GOTO 460
340 FOR I=1 TO N
350 LET D(I)=A(I)+B(I)+R : LET R=INT(D(I)/7) : LET D(I)=D(I)  - INT
(D(I)/7)*7
360 NEXT I
370 LET D(N+1)=R
380 LET D$=""
390 FOR I=1 TO N+1
400 LET D$=RIGHT$(STR$(D(I)),1)+D$
410 NEXT I
420 IF LEFT$(D$,1)<>"0" AND LEN(D$)>1 THEN GOTO 450
430 IF LEN(D$)=1 THEN GOTO 450
440 D$=RIGHT$(D$,LEN(D$) - 1) : GOTO 420
450 RETURN
460 PRINT "Suma este : ";D$ : END
470 RETURN
480 IF LEN(A$)<LEN(B$) THEN U$=" - " : GOTO 510
490 IF LEN(A$)>LEN(B$) THEN U$="" : GOTO 510
500 U$="" : IF A$<B$ THEN U$=" - "
510 R=1
520 IF U$="-" THEN GOTO 540
530 FOR I=1 TO N : B(I)=6 - B(I) : NEXT I : GOTO 550
540 FOR I=1 TO N : A(I)=6 - A(I) : NEXT I
550 GOSUB 340 : D$=U$+RIGHT$(D$,LEN(D$) - 1) : GOSUB 420
560 RETURN
```

```
570 M=LEN(A$) : IF LEN(A$)>LEN(B$) THEN M=LEN(B$) : E$=A$ :
A$=B$ : B$=E$
580 FOR I=1 TO LEN(A$)
590 A(I)=VAL(MID$(A$,LEN(A$) - I+1,1))
600 NEXT I
610 FOR I=1 TO LEN(B$)
620 B(I)=VAL(MID$(B$,LEN(B$) - I+1,1))
630 NEXT I
640 E$="0" : S$=""
650 FOR J=1 TO M
660 S$=S$+"0" : R=0
670 FOR I=1 TO N
680 D(I)=A(J)*B(I)+R : R=INT(D(I)/7) : D(I)=D(I) - 7*R
690 NEXT I
700 IF R=0 THEN H$="" : GOTO 720
710 H$=RIGHT$(STR$(R),1)
720 LET D$=""
730 FOR I=1 TO N
740 LET D$=RIGHT$(STR$(D(I)),1)+D$
750 NEXT I
760 D$=H$+D$+S$ : D$=LEFT$(D$,LEN(D$) - 1)
770 FOR I=1 TO LEN(D$)
780 D(I)=VAL(MID$(D$,LEN(D$) - I+1,1))
790 NEXT I
800 FOR I=1 TO LEN(E$)
810 E(I)=VAL(MID$(E$,LEN(E$) - I+1,1))
820 NEXT I
830 T=LEN(D$) : IF LEN(E$)>LEN(D$) THEN T=LEN(E$)
840 W=0
850 FOR I=1 TO T
860 LET F(I)=D(I)+E(I)+W : LET W=INT(F(I)/7) : LET F(I)= F(I) - INT(
F(I)/7)*7
870 NEXT I
```

```
880 LET F(T+1)=W
890 LET E$=""
900 FOR I=1 TO T+1
910 LET E$=RIGHT$(STR$(F(I)),1)+E$
920 NEXT I
930 IF LEFT$(E$,1)<>"0" AND LEN(E$)>1 THEN GOTO 960
940 IF LEN(E$)=1 THEN GOTO 960
950 E$=RIGHT$(E$,LEN(E$)-1) : GOTO 930
960 NEXT J
970 PRINT E$ : END
1190 X$=A$ : Y$=B$ : Q$=""
1200 M$=X$ : N$=Y$
1210 Q=0
1220 IF LEN(M$)<LEN(N$) THEN 1380
1230 IF LEN(M$)>LEN(N$) THEN 1290
1240 IF M$<N$ THEN 1380
1250 K1=LEN(B$)
1260 A$=M$ : FOR I=1 TO N : A(I)=0 : NEXT I : FOR I=1 TO LEN(A$) :
A(I)=VAL(MID$(A$,LEN(A$)-I+1,1)) : NEXT I
1270 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1 TO LEN(B$) :
B(I)=VAL(MID$(B$,LEN(B$)-I+1,1)) : NEXT I
1280 GOTO 1430
1290 K1=LEN(B$)
1300 A$=LEFT$(M$,LEN(B$)) : FOR I=1 TO N : A(I)=0 : NEXT I : FOR
I=1 TO LEN(A$) : A(I)=VAL(MID$(A$,LEN(A$)-I+1,1)) : NEXT I
1310 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1 TO LEN(B$) :
B(I)=VAL(MID$(B$,LEN(B$)-I+1,1)) : NEXT I
1320 IF A$<B$ THEN 1340
1330 GOTO 1430
1340 K1=LEN(B$)+1
1350 A$=LEFT$(M$,LEN(B$)+1) : FOR I=1 TO N : A(I)=0 : NEXT I : FOR
I=1 TO LEN(A$) : A(I)=VAL(MID$(A$,LEN(A$)-I+1,1)) : NEXT I
```

```
1360 B$=Y$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1 TO LEN(B$) :
B(I)=VAL(MID$(B$,LEN(B$)-I+1,1)) : NEXT I
1370 GOTO 1430
1380 LET D$=Q$
1390 IF LEFT$(D$,1)<>"0" AND LEN(D$)>1 THEN GOTO 1420
1400 IF LEN(D$)=1 THEN GOTO 1420
1410 D$=RIGHT$(D$,LEN(D$)-1) : GOTO 1150
1420 PRINT "Citul este : ";D$,"Restul este : ";A$ : END
1430 D$="0" : IF LEN(B$)=1 AND VAL(B$)=0 THEN GOTO 10
1440 M1$=B$
1450 Q=0
1460 IF LEN(A$)<LEN(B$) THEN GOTO 1540
1470 IF LEN(A$)>LEN(B$) THEN GOTO 1490
1480 IF A$<B$ THEN 1540
1490 Q=Q+1 : GOTO 1500
1500 GOSUB 480
1510 A$=D$ : FOR I=1 TO N : A(I)=0 : NEXT I : FOR I=1 TO LEN(A$) :
A(I)=VAL(MID$(A$,LEN(A$)-I+1,1)) : NEXT I
1520 B$=M1$ : FOR I=1 TO N : B(I)=0 : NEXT I : FOR I=1 TO LEN(B$)
: B(I)=VAL(MID$(B$,LEN(B$)-I+1,1)) : NEXT I
1530 GOTO 1460
1540 Q$=Q$+RIGHT$(STR$(Q),1)
1550 M$=A$+RIGHT$(M$,LEN(M$)-K1) : N$=Y$ : GOTO 1210
```

4.2. Să se determine toate numerele naturale de forma $\overline{a_1a_2\ldots a_n}$ scrise în baza 10 care sînt egale cu $a_1!+a_2!+\ldots+a_n!$.

```
10 F(1)=1 : FOR I=1 TO 9 : F(I+1)=I*F(I) : NEXT I
20 FOR I=1 TO 7 : C(I)=0 : NEXT I
30 N=1
40 FOR N1=1 TO 7*F(10)
50 FOR K=1 TO 7
60 C(K)=C(K)+1
70 IF C(K)<>10 THEN GOTO 110
```

```
80 C(K)=0
90 IF K=N THEN N=N+1
100 NEXT K
110 S=0
120 FOR K=1 TO N : S=S+F(C(K)+1) : NEXT K
130 IF S=N1 THEN PRINT S
140 NEXT N1
```

4.3. Să se rezolve următorul rebus aritmetic :

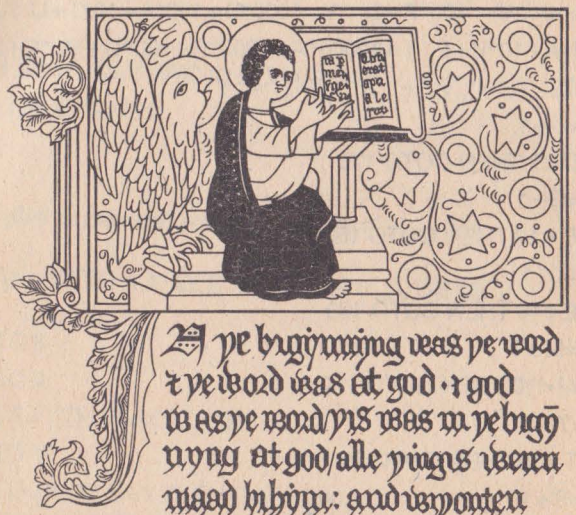
ARAD +SATU +MARE + ARGES = JUDETE

```
10 FOR A=8 TO 9
20 FOR R=0 TO 9
30 IF R=A OR R=1 THEN GOTO 360
40 FOR D=0 TO 9
50 IF D=R OR D=A OR D=1 THEN GOTO 350
60 S=2
70 IF S=D OR S=R OR S=A THEN GOTO 330
80 T=0
90 IF T=S OR T=D OR T=R OR T=A OR T=1 THEN GOTO 310
100 M=2
110 IF M=T OR M=S OR M=D OR M=R OR M=A THEN GOTO 290
120 E=0
130 IF E=M OR E=T OR E=S OR E=D OR E=R OR E=A OR E=1 THEN
GOTO 270
140 G=0
150 IF G=E OR G=M OR G=T OR G=S OR G=D OR G=R OR G=A
OR G=1 THEN GOTO 250
160 U=0
170 IF U=G OR U=E OR U=M OR U=T OR U=S OR U=D OR U=R OR
U=A OR U=1 THEN GOTO 230
180  X=A*11210+R*1110+D+S*1001+T*10+U+M*1000+ E*11+G*
100
```

```
190 Y=100000!+U*10000+D*1000+E*101+T*10
200 IF X<>Y THEN GOTO 230
210 X=1010*A+100*R+D : V=1000*S+100*A+10*T+U : Z= 1000*
M+100*A+10*R+E : T=10000*A+1000*R+100*G+10 *E+S
220 PRINT X; "+"; V; "+"; Z; "+"; T; "="; Y
230 U=U+1
240 IF U<2 THEN GOTO 170
250 G=G+1
260 IF G<10 THEN GOTO 150
270 E=E+1
280 IF E<10 THEN GOTO 130
290 M=M+1
300 IF M<10 THEN GOTO 110
310 T=T+1
320 IF T<10 THEN GOTO 90
330 S=S+1
340 IF S<10 THEN GOTO 70
350 NEXT D
360 NEXT R
370 NEXT A
```

4.4. Să se determine toate numerele naturale prime cu un număr natural *n*, mai mici decît *n*.

```
10 CLS : CLEAR : INPUT N
20 IF N<2 OR N<>INT (N) THEN GOTO 10
30 PRINT"Numerele naturale prime cu "; N ; " sint : "
40 FOR I=2 TO N
50 K=N : L=I
60 R=K - L*INT(K/L)
70 IF R=0 THEN GOTO 90
80 K=L : L=R : GOTO 60
90 IF L=1 THEN PRINT I,
100 NEXT I
```



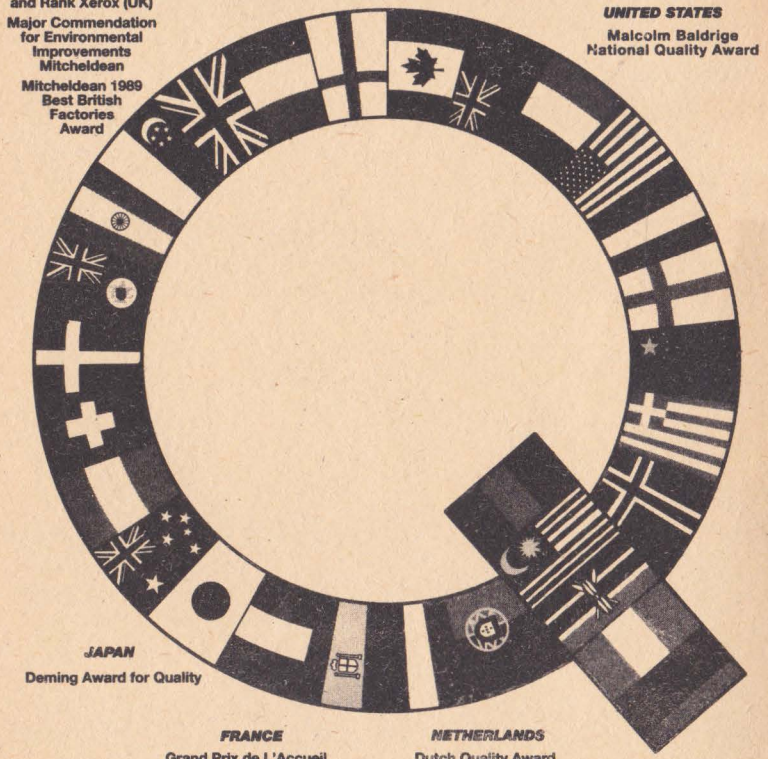
In ye bygynnyng was ye word
& ye word was at god · & god
was ye word/ys was in ye bygyn
nyng at god/alle thyngis were
maad bi hym: and with outen
hym was maad no thyng/ that
that was maad in hym was lyf·
and ye lyf was ye list of men/
and list schyneth in derknessis · and
derknessis comprehenden not it/

750
Productivity

RANK XEROX

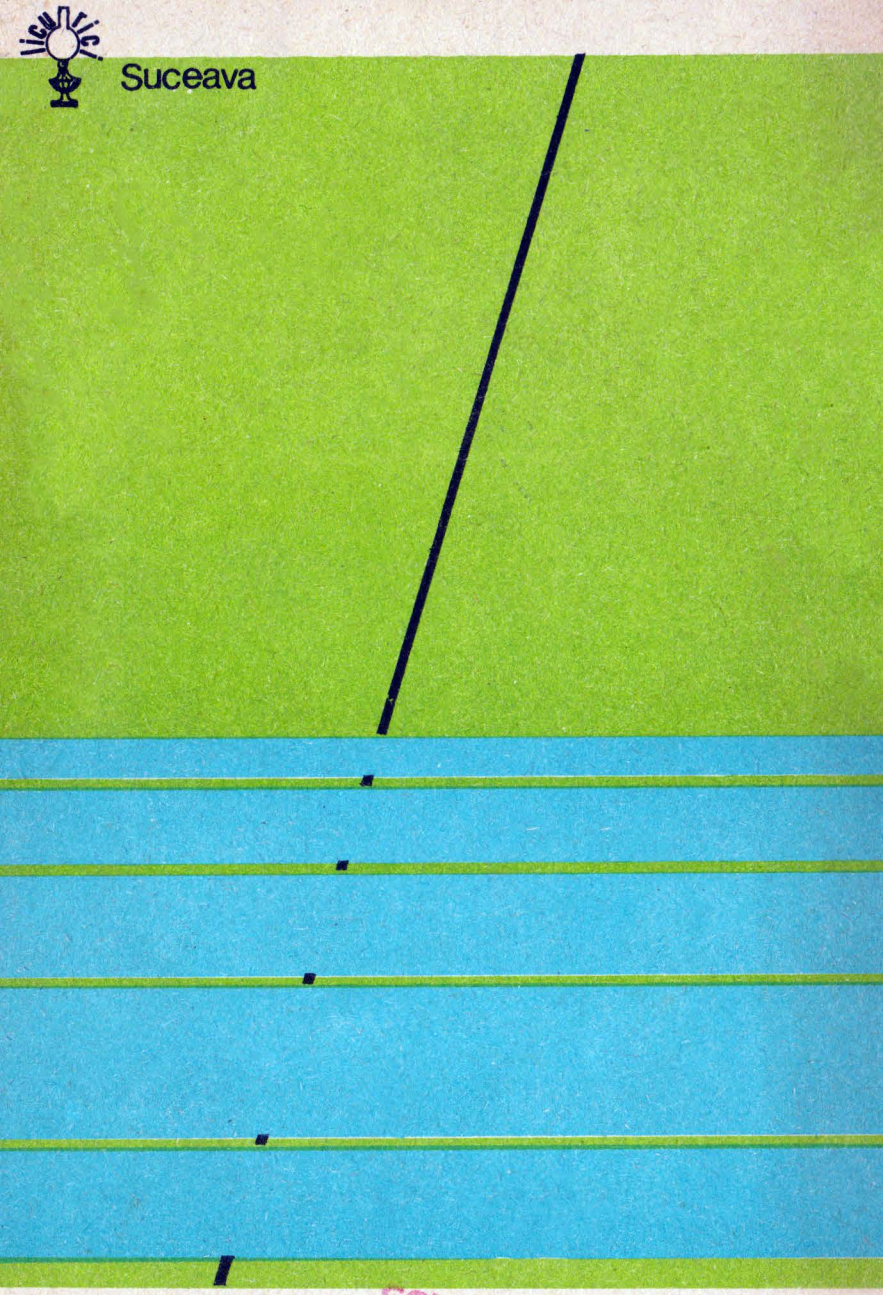
CUPRINS

S.C. „DOSOFTEI“ S.A



Quality

RANK XEROX



ISBN 973-95-73-7-0-3

lei 120