Polinoame

Capitolul

14

- Definirea noțiunii de polinom
- Forma algebrică a polinoamelor
- Reprezentarea polinoamelor în memoria calculatorului
- Implementări sugerate
- Probleme propuse
- Soluțiile problemelor

14.1. Definirea noțiunii de polinom

Fie mulțimea şirurilor (infinite) de numere complexe $f = (a_0, a_1, a_2, ..., a_n, ...)$, care au numai un număr finit de termeni nenuli, adică există un număr natural m, astfel încât $a_i = 0$ pentru orice $i > m$.

Exemple

şirul $f = (1, -2, 8, 0, ..., 0...)$ are trei termeni nenuli;
şirul $g = (0, 1, -2, 0, ..., 0)$ are doi termeni nenuli.

Pe această mulțime se definesc două operații algebrice:

1. Adunarea

$f + g = (a_0 + b_0, a_1 + b_1, a_2 + b_2, ...)$.

2. Înmulțirea

$f \cdot g = (c_0, c_1, c_2, ...)$, unde

$c_0 = a_0 \cdot b_0$

$c_1 = a_0 \cdot b_1 + a_1 \cdot b_0$

$c_2 = a_0 \cdot b_2 + a_1 \cdot b_1 + a_2 \cdot b_0$

...

$a_r = a_0 \cdot b_r + a_1 \cdot b_{r-1} + a_2 \cdot b_{r-2} +...+ a_r \cdot b_0$

Se observă că suma $f + g$ şi produsul $f \cdot g$ aparțin aceleiaşi mulțimi.

Definiție

Fiecare element al mulțimii definite anterior pe care sunt definite cele două operații, se numeşte ***polinom***.

Dacă $f = (a_0, a_1, a_2, ..., a_n,...)$ este un polinom, numerele $a_0, a_1, a_2, ...$ se numesc ***coeficienții lui f***.

A. Proprietăți ale adunării polinoamelor

Fie f, g și h trei polinoame.

1. *Adunarea este comutativă:* $f + g = g + f$.
2. *Adunarea este asociativă:* $(f + g) + h = f + (g + h)$.
3. *Element neutru* pentru adunarea polinoamelor este polinomul constant $0 = (0, 0, ...)$. Avem $f + 0 = 0 + f = f$.
4. Orice polinom are un *opus* notat cu $-f$, astfel încât $f + (-f) = (-f) + f = 0$.

B. Proprietăți ale înmulțirii polinoamelor

1. *Înmulțirea este comutativă:* $f \cdot g = g \cdot f$.
2. *Înmulțirea este asociativă:* $(f \cdot g) \cdot h = f \cdot (g \cdot h)$.
3. *Element neutru* pentru înmulțirea polinoamelor este polinomul constant $1 = (1, 0, ...)$. Avem $f \cdot 1 = 1 \cdot f = f$.
4. *Înmulțirea este distributivă față de adunare:* $f \cdot (g + h) = f \cdot g + f \cdot h$ și Avem $(f + g) \cdot h = f \cdot h + g \cdot h$.
5. Dacă f și g sunt polinoame nenule, atunci produsul lor este un polinom nenul ($f \neq 0$ și $g \neq 0 \Rightarrow f \cdot g \neq 0$).
6. Posibilitatea *simplificării* cu un factor nenul: dacă f, g și h sunt polinoame astfel încât $f \cdot g = f \cdot h$ și $f \neq 0$, atunci $g = h$.

14.2. Forma algebrică a polinoamelor

Prin convenție, vom nota polinomul (0, 1, 0, 0,...) cu X și îl vom citi *nedeterminata X*. În urma înmulțirii polinoamelor va rezulta:

$X^2 = X \cdot X = \quad (0,1,0,...) \cdot (0,1,0,0,...) = (0,0,1,0,...)$

$X^3 = X \cdot X^2 = \quad (0,1,0,...) \cdot (0,0,1,0,...) = (0,0,0,1,0,...)$

...

$$X^n = X \cdot X^{n-1} = (0,1,0,...) \cdot (\underbrace{0,0,...,0}_{n-1},1,0,...) = (\underbrace{0,0,...,0}_{n},1,0,...)$$

...

Folosim înmulțirea și adunarea pentru a scrie: $f = a_0 + a_1X + a_2X^2 + ... + a_mX^m$, unde $a_0, a_1, a_2, ..., a_m$ sunt coeficienții polinomului f.

Polinoamele de forma aX^n, unde $a \in \mathbf{C}$ (mulțimea numerelor complexe) și n este un număr natural se numesc ***monoame***.

În mulțimea polinoamelor cu coeficienți complecşi se disting următoarele submulțimi importante:
R[X] = mulțimea polinoamelor cu coeficienți reali;
Q[X] = mulțimea polinoamelor cu coeficienți raționali;
Z[X] = mulțimea polinoamelor cu coeficienți întregi.

14.3. Reprezentarea polinoamelor în memoria calculatorului

14.3.1. Reprezentarea prin şirul coeficienților

Coeficienții unui polinom se vor păstra într-un tablou unidimensional în ordine crescătoare (sau descrescătoare) după puterea lui X. Numim *gradul* polinomului puterea cea mai mare a lui X, pentru care coeficientul este diferit de 0. În concluzie, şirul corespunzător coeficienților va avea cel puțin atâtea elemente + 1 (pentru termenul liber), cât este gradul polinomului.

Acest mod de reprezentare are avantajul că în cazul a două polinoame coeficienții aceloraşi puteri ale lui X sunt aşezați în cei doi vectori pe poziții corespunzătoare.

14.3.2. Exemplu de adunare şi scădere a două polinoame

Fie $P_1(X) = 4X^5 - 3X^4 + X^2 - 8X + 1$
şi $P_2(X) = 3X^4 - X^3 + X^2 + 2X - 1$.
$(P_1 + P_2)(X) = 4X^5 - 3X^4 + X^2 - 8X + 1 + 3X^4 - X^3 + X^2 + 2X - 1 =$
$= 4X^5 - X^3 + 2X^2 - 6X.$
$(P_1 - P_2)(X) = 4X^5 - 3X^4 + X^2 - 8X + 1 - 3X^4 + X^3 - X^2 - 2X + 1 =$
$= 4X^5 - 6X^4 + X^3 - 10X + 2.$

Reprezentând polinoamele sub formă de tablouri unidimensionale în care am aşezat coeficienții în ordine crescătoare după puterile lui X, adunarea şi scăderea se efectuează astfel:

indice	0	1	2	3	4	5
P_1	1	–8	1	0	–3	4
P_2	–1	2	1	–1	3	0
$P_1 + P_2$	0	–6	2	–1	0	4
$P_1 - P_2$	2	–10	0	1	–6	4

14.3.3. Exemplu de înmulțire a două polinoame

Fie $P_1(X) = 3X^2 - X + 1$ 0 1 2 3
şi $P_2(X) = X - 2$.

$$\begin{array}{l} \underline{(3X^2 - X + 1) \cdot (X - 2)} \\ 3X^3 - X^2 + X \\ \underline{\quad -6X^2 + 2X - 2} \\ 3X^3 - 7X^2 + 3X - 2 \end{array}$$

P_1	1	–1	3	
P_2	–2	1		
$P_1 \cdot X$	0	1	–1	3
$P_1 \cdot (-2)$	–2	2	–6	
$P_1 \cdot P_2$	–2	3	–7	3

14.3.4. Valoarea unui polinom

Prin definiție numărul $f(\alpha) = a_0 + a_1\alpha + a_2\alpha^2 + ... + a_n\alpha^n$ se numeşte ***valoarea polinomului f în α***.

Exemplu
Valoarea polinomului $P_1(X) = 3X^2 - X + 1$ în punctul $X = 2$ este:
$P_1(2) = 3 \cdot 2^2 - 2 + 1 = 3 \cdot 4 - 2 + 1 = 12 - 2 + 1 = 11$.

S-a demonstrat că cea mai rapidă metodă de a calcula valoarea unui polinom este cea bazată pe schema lui *Horner*.

Polinomul P(X) se scrie sub forma:
$P(X) = (...(((a_n \cdot x + a_{n-1}) \cdot x + ... + a_2) \cdot x + a_1) \cdot x + a_0$
şi se aplică un algoritm simplu în care x este argumentul pentru care se calculează valoarea polinomului P de coeficienți a_i şi grad n:

```
Subalgoritm Horner(n,x,P):
  P ← a[n]
  pentru i=n-1,0 execută:
    P ← P*x + a[i]
sfârşit subalgoritm
```

14.3.5. Împărțirea polinoamelor

Prezentăm în continuare o serie de definiții şi teoreme a căror cunoaştere este necesară în rezolvarea problemei propuse la sfârşitul acestui capitol[*)].

A. Teorema împărțirii cu rest
Fiind date două polinoame oarecare cu coeficienții complecşi f şi g, unde $g \neq 0$, atunci există două polinoame cu coeficienți de tip complex q şi r, astfel încât $f = g \cdot q + r$, unde gradul polinomului r este mai mic decât gradul polinomului g.

Polinoamele q şi r sunt unice dacă satisfac această proprietate.

[*)] Cei care simt nevoia unui studiu aprofundat pot consulta bibliografie de specialitate (de exemplu, manualul de matematică pentru clasa a X-a).

Exemplu

Fie $P_1(X) = X^3 - 2X^2 + 6X - 5$
şi $P_2(X) = X^2 - 1$.

$$
\begin{array}{rrrrl}
(\underline{X^3} & \underline{-\,2X^2} & \underline{+\,6X} & \underline{-\,5)} & : (X^2 - 1) = X - 2 \\
\underline{-\,X^3} & & \underline{+\;X} & & \\
 & -\,2X^2 & +\,7X & -\,5 & \\
 & \underline{2X^2} & & \underline{-\,2} & \\
 & & 7X & -\,7 &
\end{array}
$$

Observaţie

Dacă cele două polinoame au coeficienţi numere întregi şi coeficientul termenului de grad maxim al împărţitorului este ± 1 atunci câtul şi restul sunt polinoame cu coeficienţi întregi.

Fie f şi g două polinoame. Spunem că ***polinomul g divide polinomul f*** (sau f este divizibil prin g, sau g este un divizor al lui f, sau f este un multiplu al lui g) dacă există un polinom h, astfel încât $f = g \cdot h$.

Fie f un polinom nenul cu coeficienţi de tip complex. Un număr complex a este ***rădăcină*** a polinomului f, dacă $f(a) = 0$.

Teorema lui Bézout:

Fie $f \neq 0$ un polinom nenul. Numărul $a \in \mathbf{C}$ ***este rădăcină a polinomului f*** dacă şi numai dacă $X - a$ îl divide pe f.

Teoremă

Fie $f = a_0 + a_1X + a_2X^2 + ... + a_nX^n$ un polinom cu $a_n \neq 0$, $n \geq 1$. Dacă $x_1, x_2, ..., x_n$ sunt rădăcinile lui f, atunci $f = a_n \cdot (X - x_1) \cdot (X - x_2) \cdot ... \cdot (X - x_n)$ şi, în plus, această descompunere a lui f în factori liniari este unică.

Teoremă

Restul împărţirii unui polinom $f \neq 0$ prin binomul $X - a$ este egal cu valoarea $f(a)$ a polinomului f în a.

Schema lui Horner oferă un procedeu de aflare a câtului şi a restului împărţirii polinomului f prin binomul $X - a$. În cazul în care restul împărţirii este 0, a este rădăcină a polinomului. Orice rădăcină a polinomului este divizor al numărului a_0/a_n (din ultima relaţie a lui *Viéte*).

În tabelul următor prezentăm *schema lui Horner* pentru împărţire:
Fie a_i coeficienţii polinomului P. Efectuăm împărţirea la monomul $X - v$.

puterile nedeterminatei	n	$n-1$	$n-2$		1	0
coeficienți	a_n	a_{n-1}	a_{n-2}	...	a_1	a_0
coeficienții câtului	a_n	$a_{n-1}+vb_{n-1}$	$a_{n-2}+vb_{n-2}$	...	a_1+vb_1	a_0+vb_0
îi notăm	b_{n-1}	b_{n-2}	b_{n-3}	...	b_0	r

Acest tabel se completează astfel:

- în rândul doi al tabelului se scriu coeficienții polinomului f;
- următoarea linie a tabelului va conține coeficienții b şi se formează astfel:
  - b_{n-1} primeşte valoarea a_n;
  - pe celelalte coloane (de la $n-1$ la 0) elementele b_i se calculează cu formula: $b_i = a_{i+1} + v \cdot b_{i+1}$
- Dacă $a_0 + vb_0$ este 0, înseamnă că v este rădăcină a polinomului. Câtul împărțirii polinomului dat la acest monom $(X-v)$ este polinomul ai cărui coeficienți sunt coeficienții b_i nenuli (doar cu un grad mai mic), iar restul este $a_0 + vb_0$.

Exemplu

Împărțim polinomul $2X^4 - 5X^3 - 8X + 1$ la $X - 2$:

puterea	v	4	3	2	1	0
coeficienți		2	–5	0	–8	1
câtul b	2	2	$-5+2\cdot 2=-1$	$0+2(-1)=-2$	$-8+2(-2)=-12$	$1+2(-12)=-23$
		b_3	b_2	b_1	b_0	r

Câtul împărțirii este $2X^3 - X^2 - 2X - 12$, iar restul este –23.

Pentru a descompune un polinom cu coeficienți întregi sub formă de produs de binoame se aplică schema lui *Horner* pentru fiecare divizor posibil al lui a_0/a_n. În cazul în care acesta este rădăcină a polinomului, se reține şi se continuă procedeul cu polinomul cât. În final, pentru fiecare rădăcină determinată avem un monom. Dacă restul este un polinom nenul înseamnă că acesta nu se poate descompune în produs de monoame în mulțimea numerelor întregi.

Exemplu

Să se descompună polinomul $X^4 - 2X^3 + 3X^2 - 10X + 8$.

Se determină rădăcinile întregi posibile, deci divizorii lui 8: 1, –1, 2, –2, 4, –4, 8, –8. Aplicăm schema lui *Horner* pentru aceste valori. Pot exista rădăcini multiple, adică putem găsi o valoare pentru care polinomul să se dividă la $(x-v)^k$, unde $k > 1$.

puterea	v	4	3	2	1	0
coeficienți		1	–2	3	–10	8
este rădăcină	**1**	1	–2 + 1 · 1 = –1	3 + (– 1) · 1 = 2	–10 + 1 · 2 = –8	8 +1 · (– 8) = **0**
dar nu dublă	1	1	–1 + 1 · 1 = 0	2 + 1 · 0 = 2	–8 + 1 · 2 = –6	
nu este rădăcină	–1	1	–1 + (– 1) · 1 = –2	2 + (–1) · (–2) = 4	–8 + (–1) · 4 = –12	
este rădăcină	**2**	1	–1 + 2 · 1 = 1	2 + 2 · 1 = 4	–8 + 2 · 4 = **0**	
nu este rădăcină	2	1	1 + 2 · 1 = 3	4 + 2 · 3 = 10		
	–2	1	1 – 2 · 1 = –1	4 + (–2) · (–1) = 6		
	4	1	1 + 1 · 4 = 5	4 + 4 · 5 = 24		
	–4	1	1 + (– 4) · 1 = –3	4 + (–4) · (–3) = 16		
	8	1	1 + 8 · 1 = 9	4 + 8 · 9 = 76		
	–8	1	1 – 8 · 1 = –7	4 + (– 8) · (– 7) = 60		

Coeficienții polinomului cât sunt:

	X^2	X^1	X^0		
	1	1	4		

În concluzie $X^4 - 2X^3 + 3X^2 - 10X + 8 = (X^2 + X + 4) \cdot (X - 1) \cdot (X - 2)$.

14.3.6. Reprezentarea polinoamelor prin monoamele sale

Dacă un polinom are mulți coeficienți egali cu 0, reprezentarea prin şirul coeficienților nu mai este avantajoasă. În acest caz, se preferă reprezentarea prin şirul monoamelor. De asemenea, dacă aceste monoame nu sunt accesibile în ordinea crescătoare (sau descrescătoare) a puterii lui X, nu vor apărea probleme, deoarece în cazul fiecărui monom se precizează două informații: valoarea coeficientului şi gradul termenului. Reprezentarea prin monoame, se poate realiza cu mai multe tipuri de structuri de date.

- cu două şiruri: unul pentru coeficienții diferiți de 0 ai polinomului şi unul pentru gradele corespunzătoare ale termenilor;
- cu un şir de articole[*)];
- cu ajutorul unei liste liniare, alocată dinamic[**)].

[*)] Tipul `record` se va învăța la sfârşitul clasei a 9-a

[**)] Alocarea dinamică o vom învăța în clasa a 10-a.

14.4. Implementări sugerate

Pentru a vă familiariza cu implementarea operațiilor cu polinoame vă recomandăm să rezolvați următoarele exerciții prin care să realizați operațiile:

1. citirea unui polinom dat prin grad şi coeficienți;
2. citirea unui polinom dat prin monoame;
3. valoarea unui polinom pentru un argument dat;
4. adunarea a două polinoame;
5. înmulțirea a două polinoame;
6. împărțirea a două polinoame;
7. determinarea acelui polinom dintre n polinoame care pentru un argument dat are valoare maximă.

14.5. Probleme propuse

14.5.1. Operații cu polinoame

În funcție de cerința precizată în fişierul de intrare, se va efectua una dintre următoarele operații asupra unor polinoame cu coeficienți întregi:

- adunarea a două polinoame;
- scăderea a două polinoame;
- înmulțirea a două polinoame;
- împărțirea a două polinoame;
- calcularea valorii unui polinom într-un punct (număr întreg) dat;
- descompunerea unui polinom în produs de binoame.

Date de intrare

De pe prima linie a fişierului de intrare **`POL.IN`** se citeşte un caracter care comunică operația care trebuie efectuată. Caracterele posibile sunt:

- `'+'`: se solicită adunarea a două polinoame;
- `'-'`: se solicită scăderea celui de-al doilea polinom din primul;
- `'*'`: se solicită produsul a două polinoame;
- `'/'`: se solicită câtul şi restul după ce primul polinom s-a împărțit la al doilea;
- `'v'`: se solicită calcularea valorii unui polinom într-un punct dat;
- `'d'`: se solicită descompunerea polinomului în binoame.

În cazul în care se doreşte o operație aritmetică, coeficienții celor două polinoame se vor afla pe liniile a doua şi respectiv a treia a fişierului.

Pentru calculul valorii unui polinom, valoarea numerică întreagă pentru care se cere calcularea valorii polinomului se va citi de pe a doua linie a fişierului, pe linia a treia aflându-se coeficienţii polinomului dat.

Dacă opţiunea este `'d'`, coeficienţii polinomului de descompus se află pe a doua linie a fişierului.

Coeficienţii polinoamelor se vor citi din fişierul **`POL.IN`** în ***ordine crescătoare după puterile nedeterminatei***. (Astfel, având şi coeficienţii nuli, gradul polinomului se va determina pe baza numărului coeficienţilor).

Date de ieşire

În fişierul **`POL.OUT`** se va scrie, în funcţie de operaţia solicitată:

- pentru operaţiile `'+'`, `'-'` şi `'*'` se va afişa polinomul rezultat în urma efectuării operaţiei respective;
- în urma operaţiei de împărţire (/) vor rezulta două polinoame (câtul şi restul); coeficienţii acestor polinoame se vor scrie pe prima (câtul) şi respectiv a doua linie a fişierului rezultat (restul);
- valoarea unui polinom cu coeficienţi întregi într-un punct dat (număr întreg) este un număr întreg care va fi scris în fişierul de ieşire;
- în urma descompunerii unui polinom ca produs de binoame rezultă o expresie de forma ($polinom_1$)($polinom_2$)...($polinom_k$).

Polinoamele se vor afişa în formă algebrică, în ordinea descrescătoare a gradelor termenilor. Puterile lui *X* se vor afişa sub forma:

– *X*`'^'`*putere* (dacă putere > 1);
– X^1 se va afişa *X*;
– X^0 nu se va afişa (apare doar coeficientul);
– termenii ai căror coeficienţi sunt 0 nu se vor scrie, excepţie făcând polinomul nul.

Restricţii şi precizări

- toate polinoamele au coeficienţi întregi;
- datele de intrare sunt corecte şi conforme cu descrierea din enunţ;
- coeficientul termenului de grad maxim a polinomului împărţitor în cazul operaţiei `'/'` este 1.

Exemple

POL.IN	**POL.OUT**

`+`
`1 -2 0 1`
`-1 -1 1`
	`X^3+X^2-3X`

POL.IN	POL.OUT

`-`
`1 -2 0 1`
`-1 -1 1`
	`X^3-X^2-X+2`

POL.IN	POL.OUT

`*`
`1 -2 0 1`
`-1 -1 1`
	`X^5-X^4-3X^3+3X^2+X-1`

POL.IN	POL.OUT

`/`
`1 -2 0 1`
`-1 -1 1`
	`X+1`
	`2`

POL.IN	POL.OUT

`v`
`5`
`1 -2 0 1`
	`116`

POL.IN	POL.OUT
`d` `8 -10 3 -2 1`	`(X^2+X+4)( X-1)( X-2)`

14.6. Soluțiile problemelor propuse

14.6.1. Operații cu polinoame

Complexitatea acestei aplicații obligă la utilizarea subprogramelor, unele dintre ele fiind necesare în cazul realizării mai multor operații. Din subprogramele care prelucrează polinoamele am construit un *unit*. În acest *unit* am definit tipul `polinom` şi subprogramele care realizează operațiile de care vom avea nevoie. Unele subprograme sunt apelate doar de subprograme şi „nu se văd” din programul utilizator al *unit*-ului.

Interfața acestui *unit*:

```
interface

type polinom=array[0..30] of Integer;

function Valp(p:polinom; gradp:Byte; x:Integer):Integer;
        { calculează valoarea polinomului p într-un punct dat x cu schema lui Horner }

procedure citeste(var pol:polinom; var gradp:Byte; var f:Text);
                                   { returnează polinomul citit din fişier şi gradul lui }
```

```
procedure Afiseaza(p:polinom; gradp:Byte; var g:Text);
                           { afişează un polinom în forma sa algebrică }

procedure Completeaza(var p:polinom; g:Byte; var grad:Byte);
                  { completează polinomul p cu 0-uri până la noul grad }

procedure Initializeaza(var p:polinom; g:Byte);
            { inițializează coeficienții polinomului p de grad g cu 0-uri }

procedure Ori_X(var p:polinom; var g:Byte);
                                      { înmulțeşte polinomul p cu X }

procedure Inmulteste_cu_scalar(var p:polinom; g:Byte;
                                        constanta:Integer);
       { returnează polinomul rezultat din înmulțirea lui cu o constantă }

procedure Aduna(p1,p2:polinom; g1,g2:Byte; var p3:polinom;
                                              var g3:Byte);
       { adunarea a două polinoame (p1 şi p2 de grad g1 respectiv g2) }

procedure Scade(p1,p2:polinom; g1,g2:Byte; var p3:polinom;
                                              var g3:Byte);
  { efectuează scăderea a două polinoame (p1 şi p2 de grad g1 respectiv g2) }

procedure Inmulteste(p1,p2:polinom; g1,g2:Byte; var p3:polinom;
                                              var g3:Byte);
       { returnează polinomul rezultat din înmulțirea a două polinoame }

procedure Imparte(p1,p2:polinom; g1,g2:Byte;
                  var pcat,prest:polinom; var gcat,grest:Byte);
  { returnează polinomul cât şi rest rezultate din împărțirea a două polinoame }

procedure Descompune(p:polinom; g:Byte; var g:Text);
                           { descompune polinomul în produs de binoame }
```

Subalgoritmul `Citeşte` citeşte de pe linia curentă a fişierului de intrare coeficienții unui polinom şi determină gradul acestuia (în funcție de numărul de coeficienți).

```
Subalgoritm Citeşte(pol,gradp):
  gradp ← 0
  cât timp nu urmează marca de sfârşit de linie execută:
    citeşte pol[gradp]
    gradp ← gradp+1
  sfârşit cât timp
  gradp ← gradp - 1
sfârşit subalgoritm
```

În subalgoritmul care afişează un polinom sub formă algebrică şi conform cerinţelor enunţate s-au separat următoarele situaţii:

- dacă polinomul are grad 0, se afişează termenul liber;
- dacă polinomul are grad mai mare decât 0:
  - se afişează primul termen (de grad maxim);
  - se afişează termenii de grad `gradp-1`, ..., `1`;
  - se afişează termenul liber.

De asemenea:

- în faţa coeficienţilor pozitivi vom scrie `'+'` (avem de afişat o sumă de monoame), pe când în faţa celor negativi semnul `'-'` se afişează implicit (nu este cazul să se adauge nimic);
- dacă coeficientul este 0, termenul respectiv nu se afişează.

```
Subalgoritm Afişează_polinom(p,gradp):
  dacă gradp = 0 atunci scrie p[gradp]      { avem numai termen liber }
  altfel
    dacă p[gradp] = -1 atunci scrie '-'          { termen de tipul „-x” }
    altfel
      dacă p[gradp] ≠ 1 atunci              { termen de tipul „Coef*x” }
        scrie p[gradp]
    sfârşit dacă
    dacă gradp > 1 atunci                   { termen de tipul „...x^k” }
      scrie 'X^',gradp)
    altfel scrie 'X'                          { termen de tipul „...x” }
    sfârşit dacă
    pentru i=gradp-1,1 execută:
      dacă p[i] ≠ 0 atunci        { dacă termenul are coeficient nenul }
        dacă p[i] = -1 atunci scrie '-'
        altfel
          dacă p[i] = 1 atunci scrie '+'
          altfel
            dacă p[i] > 1 atunci scrie '+',p[i]
            altfel scrie p[i]
            sfârşit dacă
          sfârşit dacă
        sfârşit dacă
        dacă i > 1 atunci scrie 'X^',i
                altfel scrie 'X'
        sfârşit dacă
      sfârşit dacă
    sfârşit pentru
```

```
    dacă p[0] > 0 atunci
      scrie '+',p[0]                    { afişarea termenului liber }
    sfârşit dacă
    dacă p[0] < 0 atunci
      scrie p[0]
  sfârşit dacă
sfârşit subalgoritm
```

Urmează acum prezentarea subalgoritmilor care corespund opţiunilor posibile din fişierul de intrare.

Opţiunea 'v'

```
Subalgoritm Valp(p,gradp,x)
     { calculează valoarea polinomului p într-un punct dat x cu schema lui Horner }
  v ← 0
  pentru i=gradp,0 execută:
    v ← v*x + p[i]
  sfârşit pentru
  Valp ← v
sfârşit subalgoritm
```

Opţiunea '+'

În vederea adunării a două polinoame (p_1 de grad g_1 şi p_2 de grad g_2), acestea mai întâi se aduc la aceeaşi lungime (a celui de grad maxim `grad`) cu subalgoritmul `Completează(p,g,grad)`. Acest lucru înseamnă adăugare de 0-uri până la lungimea mai mare (coeficienţi de rang mai mare), deoarece şirul coeficienţilor este ordonat crescător după puterile lui *X*. În continuare adunarea constă într-o simplă adunare a şirurilor coeficienţilor.

```
Subalgoritm Completează(p,g,grad):
                    { completează polinomul p cu 0-uri până la noul grad }
  pentru i=g+1,grad execută:
    p[i] ← 0
sfârşit subalgoritm
```

```
Subalgoritm Adună(p1,p2,g1,g2,p3,g3)
        { se completează cu 0-uri polinomul mai scurt până la lungimea celuilalt }
                                     { şi se determină gradul rezultatului }
  dacă g1 > g2 atunci
    g3 ← g1
    Completează(p2,g2,g1)
  altfel
    g3 ← g2
```

```
    Completează(p1,g1,g2)
  sfârşit dacă
  pentru i=0,g3 execută:
    p3[i] ← p1[i] + p2[i]                    { se adună termen cu termen }
                   { dacă în urma adunării celor două polinoame s-a redus gradul }
                      { polinomului rezultat, se modifică gradul polinomului sumă }
  cât timp (g3 > 0) şi (p3[g3] = 0) execută:
    g3 ← g3 - 1
  sfârşit cât timp
sfârşit subalgoritm
```

Dacă vrem să evităm această operație de completare cu 0-uri, care cauzează creşterea operațiilor elementare de adunare, putem proceda în felul următor:

```
Subalgoritm Adună(p1,p2,g1,g2,p3,g3):
                                                        { adunăm p1 cu p2 }
  dacă g1 < g2 atunci
    pentru i=0,n execută:
      p2[i] ← p2[i] + p1[i]
    sfârşit pentru
    p3 ← p2
    g3 ← g2
  altfel
    pentru i=0,n execută:
      p1[i] ← p1[i] + p2[i]
    sfârşit pentru
    p3 ← p1
    g3 ← g1
  sfârşit dacă
sfârşit subalgoritm
```

Această rezolvare are dezavantajul că îl modifică pe p_1, respectiv pe p_2, după caz. Dacă însă i-am transmite prin valoare, după revenirea din apel, i-am regăsi nemodificate.

Opțiunea `'-'`

Scăderea a două polinoame poate fi privită ca o adunare a primului polinom cu polinomul rezultat din înmulțirea celui de-al doilea polinom cu –1:
$p_1 - p_2 = p_1 + (-1) \cdot p_2$

```
Subalgoritm Scade(p1,p2,g1,g2,p3,g3):
  Înmulțeşte_cu_scalar(p2,g2,-1)
  Adună(p1,p2,g1,g2,p3,g3);
```

```
                    { dacă în urma scăderii celor două polinoame s-a redus gradul }
                   { polinomului rezultat, se modifică gradul polinomului diferenţă }
  cât timp (g3 > 0) şi (p3[g3] = 0) execută:
    g3 ← g3-1
  sfârşit cât timp
sfârşit subalgoritm
```

Opţiunea `'*'`

Înmulţirea poate fi privită ca adunarea polinoamelor obţinute din înmulţirea lui p_1 cu câte un monom de-al lui p_2. Polinomul „produs” îl vom iniţializa:

```
Subalgoritm Iniţializează(p,g):
                     { iniţializează coeficienţii polinomului p de grad g cu 0-uri }
  pentru i=0,g execută:
    p[i] ← 0
  sfârşit pentru
sfârşit subalgoritm
```

Monoamele cu care se va înmulţi p_1 le obţinem din p_2 sub formă de coeficienţi. În consecinţă ne trebuie un subalgoritm care înmulţeşte un polinom cu un număr întreg:

```
Subalgoritm Înmulţeşte_cu_scalar(p,g,constanta):
                 { returnează polinomul rezultat din înmulţirea lui cu constanta }
  pentru i=0,g execută:
    p[i] ← constanta*p[i]
  sfârşit pentru
sfârşit subalgoritm
```

În subalgoritmul `Înmulţeşte(p1,p2,g1,g2,p3,g3)` vom utiliza trei polinoame de lucru: `paux` de grad `gaux`, `termen1` şi `termen2`. Polinomul `paux` primeşte valoarea polinomului p_1, în `termen1` „strângem” polinomul rezultat (p_3), iar în `termen2` păstrăm temporar polinomul `paux` care la fiecare pas se înmulţeşte cu *X*. De asemenea, la fiecare pas (în număr de câţi termeni sunt în p_2) vom înmulţi p_1 cu coeficientul corespunzător din p_2 şi rezultatul se adună la p_3 (păstrat în `termen1`).

```
Subalgoritm Înmulţeşte(p1,p2,g1,g2,p3,g3):
                 { returnează polinomul rezultat din înmulţirea a două polinoame }
  g3 ← g1 + g2
  Iniţializează(p3,g3)                            { se iniţializează rezultatul }
  paux ← p1                                       { paux = polinom de lucru }
  gaux ← g1
               { se adună la fiecare pas polinomul paux la polinomul rezultat p3 }
```

```
  pentru i2=0,g2 execută:        { pentru fiecare coeficient al polinomului p2 }
    termen1 ← p3
    termen2 ← paux
    Înmulţeşte_cu_scalar(termen2,gaux,p2[i2])
    Adună(termen1,termen2,g3,gaux,p3,g3)
    Ori_X(paux,gaux)              { se pregăteşte paux pentru pasul următor }
  sfârşit pentru
sfârşit subalgoritm
```

Efectul subalgoritmului `Ori_X(p,g)` este „echivalent” cu înmulţirea polinomului p cu *X*. De fapt se deplasează coeficienţii în şir cu o poziţie la dreapta, eliberând astfel locul termenului liber care se iniţializează cu 0 şi în final se măreşte gradul polinomului cu 1:

```
Subalgoritm Ori_X(p,g):
                                             { înmulţeşte polinomul p cu X }
  pentru i=g,0 execută: { se deplasează coeficienţii cu o poziţie spre dreapta }
    p[i+1] ← p[i]
  sfârşit pentru
  p[0] ← 0
  g ← g + 1
sfârşit subalgoritm
```

Dacă într-o problemă am avea nevoie de un subalgoritm simplu de înmulţire, am putea folosi următorul:

```
Subalgoritm Înmulţeşte(p1,p2,g1,g2,p3,g3):
  g3 ← g1 + g2
  Iniţializează(p3,g3)                       { se iniţializează rezultatul }
  pentru i=0,g1 execută:
    pentru j=0,g2 execută:
     p3[i+j] ← p3[i+j] + p1[i]*p2[j]
    sfârşit pentru
  sfârşit pentru
sfârşit subalgoritm
```

Opţiunea `'/'`

Deoarece această aplicaţie determină câtul şi restul a două polinoame cu coeficienţi întregi, conform unui rezultat prezentat în suportul teoretic al lecţiei, coeficientul de grad maxim al polinomului împărţitor trebuie să fie 1 sau –1.

```
Subalgoritm Împarte(p1,p2,g1,g2,pcât,prest,gcât,grest):
      { returnează polinomul cât şi rest, rezultate din împărţirea a două polinoame }
  gcat ← g1 - g2{ gradul polinomului cât este diferenţa gradelor polinoamelor }
                          { se iniţializează coeficienţii polinomului pcât cu 0 }
  Iniţializează(pcât,gcât)
                         { câtul şi restul vor fi polinoame cu coeficienţi întregi }
  dacă (p2[g2] = 1) sau (p2[g2] = -1) atunci
                              { gradul deîmpărţitului ≥ gradul împărţitorului }
    cât timp g1 ≥ g2 execută:
             { coeficientul termenului de grad g1-g2 va fi rezultatul împărţirii }
      pcat[g1-g2] ← [p1[g1]/p2[g2]]
      Înmulţeşte(pcat,p2,g1-g2,g2,paux,gaux)
      Scade(p1,paux,g1,gaux,prest,grest)
      p1 ← prest          { se repetă procesul, împărţind restul actual la p2 }
      g1 ← grest
    sfârşit cât timp
  sfârşit dacă
sfârşit subalgoritm
```

Opţiunea 'd'

Pe rând, pentru fiecare posibil divizor se construieşte polinomul p_2, conform schemei lui Horner.

```
Subalgoritm Descompune(p,g,gg):
  dacă rest[p[0]/p[g]] = 0 atunci     { numai dacă are coeficienţi întregi }
    nd ← 0                      { se introduc în şirul divi divizorii lui p[0] }
    pentru i=1,Abs(p[0]) execută:
      dacă rest[p[0]/i] = 0 atunci{ i şi -i se introduc în şirul divizorilor }
        nd ← nd+1
        divi[nd] ← i
        nd ← nd+1
        divi[nd] ← -i
      sfârşit dacă
    sfârşit pentru
    grad ← g      { variabilă auxiliară în care se păstrează gradul polinomului }
    k ← 1                                      { indice în şirul divizorilor }
    nrăd ← 0                                 { numărul rădăcinilor găsite }
    repetă          { se construieşte polinomul p2 conform schemei lui Horner, }
      p2[g] ← p[g]                    { coeficientul termenului de grad maxim }
      pentru i=g-1,0 execută:
        p2[i] ← p[i] + p2[i+1]*divi[k]
      sfârşit pentru
```

```
      dacă p2[0] = 0 atunci             { divi[k] este rădăcină a lui p1 }
        nrăd ← nrăd + 1
        răd[nrăd] ← divi[k]         { păstrăm rădăcina în şirul rădăcinilor }
        pentru i=1,g execută:
           p[i-1] ← p2[i]
        sfârşit pentru
        g ← g - 1                          { scade gradul polinomului p1 }
      altfel
        k ← k + 1
      sfârşit dacă
    până când (k > nd) sau (nrăd = grad)
                 { până când nu mai sunt divizori sau am găsit toate rădăcinile }
    Afişează_Produs_de_Binoame
  sfârşit dacă
sfârşit subalgoritm
```

Subalgoritmul de mai sus apelează o procedură specială de afişare. Procedura face parte din zona de declaraţii a subprogramului `Descompune(p,g,gg)`.

```
Subalgoritm Afişează_Produs_de_Binoame:
                              { se afişează produsul de polinoame determinat }
  dacă g > 0 atunci
         { se afişează p1 care nu mai are rădăcini în mulţimea numerelor întregi }
    scrie '('
    Afişează(p,g,gg)
    scrie ')'
  sfârşit dacă
  pentru k=1,nrad execută:
    p[0] ← -răd[k]
    p[1] ← 1
    scrie '('
    Afişează(p,1,gg)
    scrie ')'
  sfârşit pentru
sfârşit algoritm
```

Aceste subprograme sunt toate încorporate în *unit*-ul păstrat în fişierul `Poli.pas`, care trebuie inclus în programul care îl va utiliza. Programul principal cuprinde citirea caracterului care codifică operaţia de efectuat. În funcţie de valoarea acestui caracter se execută secvenţa de instrucţiuni care citeşte datele de intrare, efectuează operaţia precizată şi afişează rezultatele în forma solicitată. `p1`, `p2`, `p3` şi `p4` sunt polinoame de grade `g1`, `g2`, `g3` şi respectiv `g4`.