SUMA puterilor ASEMENEA

Doina Hrinciuc-Logofătu

Ne propunem prezentarea unui algoritm de tip Programare Dinamică pentru problema puterilor asemenea a primelor n numere naturale. Primul pas va fi codificarea problemei de rezolvat, adică stabilirea unui suport matematic care să permită transformarea progresivă a datelor de intrare în soluție. Vor urma descrierea algoritmului, analiza sa, motivele pentru încadrarea lui în clasa problemelor de tip programare dinamică, prezentarea generală a acestei metode și două probleme propuse.

Problema sumei puterilor asemenea

Fie $k \geq 0$ un număr natural; notăm:

$$S_k(n) = 1^k + 2^k + 3^k + \ldots + n^k \qquad (1)$$

Se poate arăta că suma $S_k(n)$ este un polinom de grad $k + 1$, în care variabila n are coeficienți raționali, adică:

$$S_k(n) = \frac{1}{M}\left(a_{k+1}n^{k+1} + a_k n^k + \cdots + a_1 n + a_0\right) \qquad (2)$$

Numerele M, a_{k+1}, a_k, ..., a_1, a_0 sunt întregi, iar numărul întreg M trebuie să fie pozitiv și cât mai mic posibil.

Impunând această condiție, întreaga secvență (M, a_{k+1}, a_k, ..., a_1, a_0) va fi unică pentru numărul natural dat k.

Problema constă în determinarea secvenței (M, a_{k+1}, a_k, ..., a_1, a_0) pentru un număr natural k dat.

Date de intrare

Fișierul de intrare **`suma.in`** conține un singur număr natural k ($0 \leq k \leq 20$).

Date de ieșire

Fișierul de ieșire **`suma.out`** trebuie să conțină valorile M, a_{k+1}, a_k, ..., a_1, a_0, în această ordine și separate prin unul sau mai multe spații.

Exemplu

`suma.in`

```
2
5
0
```

`suma.out`

```
6 2 3 1 0
12 2 6 5 0 -1 0 0
1 1 0
```

Analiza problemei. Modelarea algebrică

Prezentăm în continuare câteva dintre formulele relativ cunoscute, care pot fi aplicate pentru valori mici ale lui k.

- $S_0(n) = \sum_{i=1}^{n} i^0 = n \qquad (3)$

- $S_1(n) = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{1}{2}\left(n^2 + n\right) \qquad (4)$

- $S_2(n) = \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} = \frac{1}{6}\left(2n^3 + 3n^2 + n\right) \qquad (5)$

- $S_3(n) = \sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4} = \frac{1}{4}\left(n^4 + 2n^3 + n^2\right) \qquad (6)$

- $S_4(n) = \sum_{i=1}^{n} i^4 = \frac{n(n+)1(2n+1)(3n^2+3n-1)}{30} = \frac{1}{30}\left(6n^5 + 15n^4 + 10n^3 - n\right) \qquad (7)$

Ne propunem în continuare determinarea unei formule pentru $S_5(n)$. Pe baza formulei lui *Newton*, scriem expresia binomială:

$$(i-1)^6 = i^6 - C_6^1 i^5 + C_6^2 i^4 - C_6^3 i^3 + C_6^4 i^2 - C_6^5 i + 1 \qquad (8)$$

care este echivalentă cu:

$$i^6 - (i-)1^6 = 6i^5 - 15i^4 + 20i^3 - 15i^2 + 6i - 1 \qquad (9)$$

Aplicând suma pentru i de la 1 la n, vom obține:

$$\sum_{i=1}^{n}\left(i^6-(i-1)^6\right)=$$
$$=6\sum_{i=1}^{n}i^5-15\sum_{i=1}^{n}i^4+20\sum_{i=1}^{n}i^3-$$
$$-15\sum_{k=1}^{n}i^2+6\sum_{i=1}^{n}i-\sum_{i=1}^{n}1 \qquad (10)$$

După reducerea termenilor din partea stângă, obținem:

$$n^6=6S_5(n)-15S_4(n)+20S_3(n)$$
$$-15S_2(n)+6S_1(n)-S_0(n) \qquad (11)$$

Rezolvând ecuația **(11)** pentru a afla $S_5(n)$, prin înlocuirea expresiilor $S_4(n)$, $S_3(n)$, ..., $S_0(n)$, obținem:

$$S_5(n)=\frac{n^2(n+1)^2\left(2n^2+2n-1\right)}{12}$$
$$=\frac{1}{12}\left(2n^6+6n^5+5n^4-n^2\right) \qquad (12)$$

Prin extrapolarea acestui exemplu concret, putem scrie în general:

$$(k+1)S_k(n)=n^{k+1}+C_{k+1}^2S_{k-1}(n)$$
$$-C_{k+1}^3S_{k-2}(n)+\cdots$$
$$+(-1)^pC_{k+1}^pS_{k-p+1}(n)\cdot$$
$$+(-1)^kC_{k+1}^kS_1(n) \qquad (13)$$
$$+(-1)^{k+1}S_0(n)$$

Observăm că pentru calcularea valorii $S_k(n)$ sunt necesare toate sumele asemenea de putere mai mică decât k.

De la ecuația (13) spre algoritm

Revenind la problema noastră, deducem că pentru aflarea secvenței (M, a_{k+1}, a_k, ..., a_1, a_0) pentru un număr natural dat k sunt necesare toate secvențele corespunzătoare pentru toate sumele cu grad mai mic decât k, începând cu cea de grad 0.

Așadar, toate aceste informații trebuie salvate progresiv pentru a putea fi manipulate în procesul de determinare a soluției pentru instanța k.

Vom considera un tablou unidimensional `M[]` în care vor fi salvate secvențial valorile `M[0]`, `M[2]`,..., `M[k]` cu proprietatea: `M[i]` *este numărul întreg pozitiv ce satisface soluția problemei pentru* $S_i(n)$, $0 \le i \le k$.

De asemenea, vom considera un tablou bidimensional `A[][]` în care vom salva valorile (a_{i+1}, a_i, ..., a_1, a_0) pentru $0 \le i \le k$, cu semnificația: *linia* i *a tabloului* `A[][]` *conține coeficienții corespunzători sumei* $S_i(n)$.

Folosind aceste reprezentări și înlocuind în formula (2), vom putea scrie:

$$S_k(n)=\frac{1}{M[k]}\left(A[k][k+1]n^{k+1}+\right.$$
$$+A[k][k]n^k+\cdots+ \qquad (14)$$
$$\left.+A[k][1]n+A[k][0]\right)$$

În continuare vom construi formule recurente pentru secvența ($M[k]$, $A[k][k+1]$, $A[k][k]$, ..., $A[k][1]$, $A[k][0]$) pe baza secvențelor deja cunoscute ($M[i]$, $A[i][i+1]$, $A[i][i]$, ..., $A[i][1]$, $A[i][0]$) pentru $0 \le i < k$.

Acest lucru pare anevoios la prima vedere, dar vom vedea cum simpla "punere pe hârtie" a ecuației (13) cu această notație ne va furniza informații prețioase și alte recurențe utile.

Vom obține succesiv:

$$(k+1)S_k(n)=n^{k+1}$$
$$+C_{k+1}^2\frac{1}{M[k-1]}\left(A[k-1][k]n^k+\right.$$
$$+A[k-1][k-1]n^{k-1}+\ldots+$$
$$\left.+A[k-1][1]n+A[k-1][0]\right)-$$
$$-C_{k+1}^3\frac{1}{M[k-1]}\left(A[k-2][k-1]n^k+\right.$$
$$+A[k-2][k-2]n^{k-1}+\ldots+$$
$$\left.+A[k-2][1]n+A[k-2][0]\right)+$$
$$+\cdots+$$
$$+(-1)^rC_{k+1}^r\frac{1}{M[k-1]}($$
$$A[k+1-r][k+2-r]n^{k-1}$$
$$+A[k+1-r][k+1-r]n^{k-1}$$
$$+\ldots+$$
$$+A[k+1-r][1]n+$$
$$+A[k+1-r][0])+$$
$$+\cdots+$$
$$+(-1)^kC_{k+1}^k\frac{1}{M[1]}\left(A[1][2]n^2+\right.$$
$$\left.+A[1][1]n+A[1][0]\right)+$$
$$+(-1)^{k+1}C_{k+1}^{k+1}\frac{1}{M[0]}\left(A[0][1]n+\right. \qquad (15)$$
$$\left.+A[0][0]\right)$$

Ecuația a fost obținută prin înlocuirea sumelor puterilor din formula (13) folosind notația cu tablouri.

În continuare, vom rescrie ecuația (15) sub forma unui polinom de grad $k+1$ grupând termenii după puterile lui n.

$$(k+1)S_k(n)=n^{k+1}+$$
$$+n^k\frac{C_{k+1}^2}{M[k-1]}A[k-1][k]+$$
$$+n^{k-1}\left(\frac{C_{k+1}^2}{M[k-1]}A[k-1][k-1]-\right.$$
$$\left.-\frac{C_{k+1}^3}{M[k-2]}A[k-2][k-1]\right)+$$
$$+n^{k-2}\left(\frac{C_{k+1}^2}{M[k-1]}A[k-1][k-2]-\right.$$
$$-\frac{C_{k+1}^3}{M[k-2]}A[k-2][k-2]+$$
$$\left.+\frac{C_{k+1}^4}{M[k-3]}A[k-3][k-2]\right)+$$
$$+\cdots+$$
$$+n^r\left(\frac{C_{k+1}^2}{M[k-1]}A[k-1][r]-\right.$$
$$-\frac{C_{k+1}^3}{M[k-2]}A[k-2][r]+$$
$$+\cdots+$$
$$\left.+(-1)^{k+2-r}\frac{C_{k+1}^{k+2-r}}{M[r-1]}A[r-1][r]\right)$$
$$+\cdots+$$
$$+n\left(\frac{C_{k+1}^2}{M[k-1]}A[k-1][1]-\right.$$
$$-\frac{C_{k+1}^3}{M[k-2]}A[k-2][1]+$$
$$+\cdots+$$
$$\left.+(-1)^{k+1}\frac{C_{k+1}^{k+1}}{M[0]}A[0][1]\right)+$$
$$+\left(\frac{C_{k+1}^2}{M[k-1]}A[k-1][0]-\right.$$
$$-\frac{C_{k+1}^3}{M[k-2]}A[k-2][0]+$$
$$+\cdots+$$
$$\left.+(-1)^{k+1}\frac{C_{k+1}^{k+1}}{M[0]}A[0][0]\right) \qquad (16)$$

Privind cu atenție ecuația **(16)**, se remarcă faptul că toți coeficienții elementelor din `A[][]` sunt aceleași funcții raționale, ce pot fi scrise sub forma:

$$W[t]=(-1)^{k+1-t}\frac{C_{k+1}^{k+1-t}}{M[t]},0\le t\le k. \qquad (17)$$

Folosind ecuația **(17)**, ecuația **(16)** devine:

$$
\begin{aligned}
(k+1)S_k(n) &= n^{k+1} + \\
&n^k(W[k-1]A[k-1][k])+ \\
&n^{k-1}(W[k-1]A[k-1][k-1]+ \\
&\quad +W[k-2]A[k-2][k-1])+ \\
&n^{k-2}(W[k-1]A[k-1][k-2]+ \\
&\quad +W[k-2]A[k-2][k-2]+ \\
&\quad +W[k-3]A[k-3][k-2])+ \\
&+\cdots+ \\
&n^r(W[k-1]A[k-1][r]+ \\
&\quad +W[k-2]A[k-2][r]+ \\
&\quad +\cdots+ \\
&\quad +W[r-1]A[r-1][r]) \\
&+\cdots+ \\
&n(W[k-1]A[k-1][1]+ \\
&\quad +W[k-2]A[k-2][1]+ \\
&\quad +\cdots+ \\
&\quad +W[0]A[0][1])+ \\
&+(W[k-1]A[k-1][0]+ \\
&\quad +W[k-2]A[k-2][0]+ \\
&\quad +\cdots+ \\
&\quad +W[0]A[0][0])
\end{aligned} \tag{18}
$$

Cu aceasta situația s-a simplificat mult. Coeficienții puterilor lui n din această ecuație se pot stoca într-un tablou unidimensional de fracții rationale `F[]` și au forma:

$$
\begin{aligned}
F[0] &= \sum_{t=k-1}^{1} W[t]A[t][0] + W[0]A[0][0] \\
F[r] &= \sum_{t=k-1}^{p-1} W[t]A[t][r],\ 1 \le r \le k \\
F[k+1] &= 1
\end{aligned} \tag{19}
$$

Înlocuind în ecuația **(18)** valorile din ecuația **(19)** obținem:

$$
\begin{aligned}
S_k(n) = \frac{1}{k+1}(&F[k+1]n^{k+1} + F[k]n^k + \\
&+ F[k-1]n^{k-1} + \cdots + F[1]n + F[0]
\end{aligned} \tag{20}
$$

Am ajuns, succesiv la o formulă asemănătoare cu cerința problemei. Mai rămâne acum aducerea în forma cerută cu M minimal.

Aceasta se realizează prin aflarea celui mai mic multiplu comun a numitorilor fracțiilor $F[0]$, $F[1]$, …, $F[k]$, $F[k+1]$. Notând acest multiplu prin Q, putem rescrie ecuația (20) astfel:

$$
\begin{aligned}
S_k(n) &= \frac{1}{(k+1)Q}(F'[k+1]n^{k+1} + \\
&\quad + F'[k]n^k + F'[k]n^{k-1} + \cdots + \\
&\quad + F'[1]n + F'[0]) \\
F'[r] &= F[r] \cdot Q,\ 0 \le r \le k+1
\end{aligned} \tag{21}
$$

Deoarece Q este cel mai mic multiplu comun al numitorilor fracțiilor `F[]`, `F'[]` va conține numere naturale și $(k+1) \cdot Q$ este cel mai mic M pentru formula $S_k(n)$ în forma cerută.

Algoritmul

Deoarece $S_0(n) = \sum_{i=1}^{n} i^0 = n$, deducem că $M[0] = 1$, $A[0][1] = 1$, $A[0][0] = 0$.

Ținând cont de manipulările algebrice anterioare, algoritmul de rezolvare al problemei va fi următorul:

```
algoritm SumaPuteriAsemenea(k)
1. citește k
2. GenerazăCombinări(
                    C[k+1][k+1])
3. M[0] ← 1
   A[0][1] ← 1
   A[0][0] ← 0
4. pentru p ← 1, k execută
4.1. semn ← 1
4.2. pentru t ← p - 1, 0 pas -1
                          execută
       W[t] ←
         FracțieSimplificată(
          semn · C[p+1][p+1-t]
                       / M[t] )
       semn ← -1 · semn * (-1)
     sfârșit pentru
4.3. pentru t ← p, 1 pas -1
                          execută
       fAux ← Fracție(0/1)
       pentru r ← p - 1, t - 1
          pas -1, r ≥ 0 execută
          fAux2 ←
           FracțieSimplificată
                (W[r] · A[r][t])
          fAux ← SumaFracții
                 (fAux, fAux2)
       sfârșit pentru
       F[t] ← fAux
     sfârșit pentru
4.4. fAux ←
          FracțieSimplificată
               (W[0] · A[0][0])
     F[0] ← SumaFracții
                 (F[0], fAux)
4.5. F[p+1] ← Fracție(1/1)
4.6. Q ← CMMMCNumitori
                  (F[], p + 1)
4.7. M[p] ← (p + 1) · Q
4.8. pentru t ← p + 1, 0 pas -1
                          execută
       fAux2 ←
          FracțieSimplificată
                    (F[t] · Q)
       A[p][t] ← Numărător
                    (fAux)
     sfârșit pentru
   sfârșit pentru
5. scrie M[k]
   pentru i ← k + 1, 0 pas -1
                          execută
     scrie A[k][i]
   sfârșit pentru
sfârșit algoritm
```

Descrierea algoritmului

La pasul `2.` se generează toți coeficienții binomiali până la ordinul $k + 1$ (conform ecuației **(13)**, pentru calcularea valorii $S_k(n)$ sunt necesari coeficienții binomiali de ordin $k + 1$).

Acesta se poate realiza, tot utilizând un algoritm bazat pe metoda programării dinamice, prin calcularea succesivă a valorilor $C[i][j]$ folosind formulele:

$$
\begin{aligned}
C[i][0] = C[i][i] &= 1,\ 0 \le i \le k+1 \\
C[i][j] = C[i-1][j-1] &+ C[i-1][j], \\
&1 \le i \le k+1,\ 1 \le j < i
\end{aligned}
$$

La pasul `3.` se inițializează valorile corespunzătoare valorilor $S_0(n)$ (conform ecuațiilor **(2)** și **(14)**).

La pasul `4.` se execută o instrucțiune repetitivă, care determină progresiv valorile $A[p][0]$, $A[p][1]$, ..., $A[p][p+1]$, $M[p]$, $1 \le p \le k$. Acestea reprezintă soluțiile pentru problemele de ordinul 1, 2, ..., k.

La pașii `4.1.` și `4.2.` se calculează fracțiile `W[]` conform ecuației **(17)**. Remarcăm folosirea unei metode `FuncțieSimplificată()` care trebuie să returneze valoarea ireductibilă a unei fracții. Aceasta se realizează prin determinarea celui mai mare divizor comun d a numitorului și numărătorului, urmată de simplificarea lor cu d.

La pasul `4.3.` se vor calcula fracțiile ireductibile `F[]` utilizând ecuațiile **(19)**. Remarcăm folosirea metodei `SumaFracții(F1, F2)` care returnează fracția ireductibilă reprezentând suma a două fracții raționale.

La pasul `4.4.` se adaugă la $F[0]$ valoarea $W[0] \cdot A[0][0]$, în conformitate cu prima parte a ecuațiilor (19).

Tot conform ecuațiilor (19), la pasul `4.5.` se atribuie valorii $F[p+1]$ fracția 1/1. La pasul `4.6.` se atribuie valorii Q cel mai mic multiplu comun al numitorilor fracțiilor $F[0]$, $F[1]$, ..., $F[p]$, $F[p+1]$ folosind o metodă `CMMMCNumitori()`.

La pasul `4.7.` se setează valoarea $M[p]$ conform ecuației **(21)**. Tot conform ecuației **(21)** se calculează la pasul `4.8.` valorile $A[p][p+1]$, $A[p][p]$, …, $A[p][0]$.

Analiza algoritmului

Ordinul de complexitate al algoritmului este $O(n^3)$ (pasul `4.` conține trei instrucțiuni repetitive imbricate), așadar algoritmul este polinomial.

Corectitudinea

Algoritmul este corect, deoarece se bazează pe succesiuni corecte de manipulări matematice.

Vom nota cu $P(k)$ problema determinării sumei pentru puterile k. Constatăm că, în cadrul algoritmului nostru, rezolvăm pe rând problemele $P(0)$, $P(1)$, ..., $P(k-1)$.

Pe baza tuturor soluțiilor acestor subprobleme se ***decid*** valorile pentru problema $P(k)$:

$$SOL\_P(k) = FUNCȚIE(SOL\_P(0), SOL\_P(1), ..., SOL\_P(k-1))$$

Soluția unei probleme $P(i)$ participă la rezolvarea tuturor problemelor $P(i+1)$, $P(i+2)$, …, motiv pentru care soluția unei probleme $P(i)$ este construită o singură dată și salvată pentru a fi utililizată ulterior repetat de problemele de dimensiuni mai mari.

Acesta este un algoritm clasic în care este utilizată metoda programării dinamice.

Dependența

Fiecare element a_i pentru o valoare k dată depinde de valorile a_i (dacă acestea există) corespunzătoare valorilor $k-1$, $k-2$, ...; valoarea M depinde de toate aceste elemente, fiind un factor de normalizare al formulei.

Considerații generale asupra programării dinamice

Metoda constă în determinarea soluției problemei pe baza unui șir de decizii d_1, d_2, ..., d_n unde d_i transformă problema din starea s_{i-1} în starea s_i.

Spre deosebire de metoda *divide et impera*, unde subproblemele trebuie să fie independente, în cadrul metodei programării dinamice, subproblemele vor fi subprobleme comune mai multor subprobleme de dimensiuni mai mari.

Pentru a fi eficientă, metoda programării dinamice trebuie să rezolve fiecare subproblemă o singură dată și să memoreze soluția pentru a o utiliza în mai multe subprobleme ulterioare.

Spre deosebire de metoda *divide et impera*, metoda programării dinamice operează de jos în sus, nerecursiv și presupune cunoașterea de la început a subproblemelor care apar în descompunerea unei probleme.

Metoda programării dinamice nu este o tehnică standardizată cum este, de exemplu, metoda *backtracking*, iar elaborarea algoritmului se bazează pe câteva principii generale.

Din acest motiv, abordarea acestei metode nu este evidentă și presupune aptitudini de programare, deseori dublate de aptitudini matematice.

Deoarece metoda programării dinamice determină o soluție fără a exploata toate cazurile posibile, putem găsi o asemănare și cu metoda *greedy*, diferența constând în faptul că în cadrul metodei *greedy* se alege optimul local la un moment dat, iar în cazul programării dinamice se exploatează proprietatea de substructură optimă a problemei (combinarea soluțiilor optime ale subproblemelor).

Probleme propuse

Problema #1

Considerăm sumele puterilor asemenea, scrise sub forma

$$S_k(n) = F_{k+1}n^{k+1} + F_k n^k + ... + F_1 n + F_0,$$

unde F_{k+1}, F_k, ..., F_0 sunt fracții raționale ireductibile.

Să se determine secvența (F_{k+1}, F_k, ..., F_1, F_0) pentru un număr natural dat.

Modelați soluția problemei iterativ, ca mai sus, utilizând aceasta formă simplificată.

Problema #2

Se consideră n numere naturale a_1, a_2, ..., a_n ($n < 30$) mai mici decât 100. Să se determine o listă de numere b_1, b_2, ..., b_n, care satisfac relația:

$$b_i = \sum_{(k_1,k_2,...,k_i)} a_{k_1} \cdot a_{k_2} \cdot \cdots \cdot a_{k_i},$$
$$1 \le k_1 < k_2 < ... < k_i \le n$$

De exemplu, pentru n = *3* se va obține:

$b_1 = a_1 + a_2 + a_3$
$b_2 = a_1 \cdot a_2 + a_1 \cdot a_3 + a_2 \cdot a_3$
$b_3 = a_1 \cdot a_2 \cdot a_3$

Indicație

Se vor utiliza relațiile lui *Viete* pentru rădăcinile ecuației $x_n + a_{n-1} \cdot x_{n-1} + a_{n-2} \cdot x_{n-2} + \ldots + a_0 = 0$, care poate fi scrisă și sub forma : $(x - x_1) \cdot (x - x_2) \cdot ... \cdot (x - x_n) = 0$ pentru construcția iterativă a polinomului.

De exemplu, pentru a_k avem $(X - a[k]) \cdot (P[0] + P[1] \cdot X + ... + P[k-1] \cdot X^{k-1} + P[k] \cdot X^k)$. Inițial vom avea $P[0] = -a[0]$ și $P[1] = 1$, iar pentru k de la 1 la $n-1$, sunt adevărate relațiile:
$P[0] \leftarrow -P[0] \cdot a[k]$;
$P[j] \leftarrow P[j-1] - P[j] \cdot a[k]$,
pentru j cuprins între 1 și k;
$P[k+1] \leftarrow P[k]$.

Bibliografie:

1. *ACM North-Eastern European Regional Programming*, Problem B, 1997-1998
2. *ACM South-Eastern European Regional Programming*, Problem C, 1999
3. Graham R. L., Knuth D. E., Patashnik O., *Concrete Mathematics*, Addison-Wesley, 1994
4. Hrinciuc Logofătu D., *C++. Probleme rezolvate și algoritmi*, editura Polirom, Iași, 2001
5. Năstăsescu C., Niță C., Popa S., *Algebră, manual pentru clasa a X-a*, Editura Didactică și Pedagogică, București, 1997