POLYGON
VII. kötet2. szám1997.
85-88.

EGY ÖTLET

Számrendszerek alkalmazásáról (Egy gondolat Pintér Klára cikke nyomán)

SZALKAI ISTVÁN

Érdeklődéssel olvastam Pintér Klára [1] cikkét jelen folyóirat hasábjain, melyben a nem-tízes alapú számrendszerek több alkalmazását élvezetesen mutatja be feladatokon keresztül. Az alábbiakban még két alkalmazást szeretnék megismertetni röviden, melyeknek közös tanulsága — no, erről majd a cikk végén (pontosabban a 2. Feladat után).

Varázsdiók az asztalon. Kezdjük az alábbi játékkal, melyet kisfiam talált Perelman [2] könyvében:

Vegyünk elő három kisméretű tárgyat, mondjuk ceruza, kulcs, bicska. Az asztalra teszünk 24 diót, amiből három önként jelentkezőnek 1, 2, *illetve* 3 *szemet adunk. Megkérjük őket, hogy távollétünkben rejtsenek el zsebükbe egy-egy tárgyat. Továbbá: aki a ceruzát dugta el, vegyen el még ugyanannyi diót, mint amennyit adtunk neki; a kulcs gazdája kétszer annyit vegyen el (mint amennyit tőlünk kapott), míg a bicskánkat rejtegető dióinak négyszeresét vegye még el. A szobába visszatérve rápillantunk az asztalon levő diókra (megszámláljuk őket). és kitaláljuk, kinél milyen tárgy van!*

A megoldás kulcsát, azt hiszem, nem kell részleteznem: minden lehetőség (permutáció) esetén más az asztalon maradó diók száma. Ezt kis táblázatban könnyen ki is mutathatjuk, sőt hatásosabb a mutatvány, ha a táblázatot kívülről is megtanuljuk!

A trükk általánosítása a következőt követeli meg: rögzített (tetszőleges) $n \in \mathbb{N}$ természetes szám esetén keresünk olyan $x_1, x_2 \ldots, x_n \in \mathbb{N}$ és $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{N}$ természetes számokat, amelyekre a

$$T_\pi := \sum_{i=1}^{n} \alpha_{\pi(i)} \cdot x_i$$

összegek az $1, 2, \ldots, n$ számok minden π permutációjára különbözőek! Itt természetesen $x_1, x_2, \ldots, x_n$ a játék kezdetén kiosztott diók száma, és $\alpha_1, \alpha_2, \ldots, \alpha_n$ a szorzótényezők az eldugott tárgyaknak megfelelően. A bevezetőben ismertetett példában $n = 3$, $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $\alpha_1 = 1$, $\alpha_2 = 2$ és $\alpha_3 = 4$.

Vagyis feladatunk: tetszőleges n természetes számhoz kell keresni fenti tulajdonságú α_i és x_i természetes számokat, lehetőleg minél kisebbeket.

Megoldásunk ötlete a következő: adjunk olyan utasításokat a játékosoknak (azaz válasszunk olyan α_i és x_i számokat), hogy az asztalról elvett diók számában, ha azt n-alapú számrendszerben írjuk fel, az egyes helyiértékeken álló *számjegyek* adják meg (kódolják) azt, hogy az egyes tárgyakat kik rejtették el. Vagyis legyen

$$x_i := i \quad \text{és} \quad \alpha_i = n^{i-1}, \quad \text{ha} \quad 1 \leq i \leq n.$$

Első ránézésre túl soknak gondoljuk a felhasznált diók számát: $n^n - 1$. Hát még ha észrevesszük azt is, hogy ezzel a módszerrel még azon eseteket is kódolhatjuk az asztalról elvett diók számában, amikor a játékosok azonos tárgyakat is elrejthetnek egymástól függetlenül. Sajnos azonban nagyságrendekkel kevesebb dióval biztosan nem boldogulhatunk, hiszen az eredeti feladatban a tárgyak lehetséges elrejtéseinek száma $n!$, ami a Stirling-formula szerint közelítőleg $\left(\frac{n}{e}\right)^n \cdot \sqrt{2\pi n}$, és láthatóan alig kisebb, mint az általunk elért n^{n-1}. Vagyis (körülbelül) ennyi dió mindenképpen kell a mutatványhoz!

Nem ismerünk a közölt módszernél jobb, azaz $n^n - 1$ diónál lényegesen kevesebb diót használó módszert, az Olvasók eredményeit izgatottan várjuk!

1. Feladat*. *Tetszőleges n szám esetén adjunk a közölt módszernél (lényegesen) kevesebb diót felhasználó módszert, azaz adjuk meg az x_i és α_i $(1 \leq i \leq n)$ számokat!*

2. FOR-ciklusok. Ha egy számítógép programban *előre* adott számú For-Next ciklust kell egymásba ágyaznunk, mi sem egyszerűbb:

```
For i_1 := 0 to n − 1
    For i_2 := 0 to n − 1
        ...
        For i_k := 0 to n − 1
        ··· utasítások ···
        Next i_k
        ...
    Next i_2
Next i_1
```

No persze k-t már a program írásakor előre ismernünk kell!

Mit teszünk akkor, ha már k-t (az egymásba zárt ciklusok számát) is csak a program futásakor ismerjük meg, inputként? Hogyan tudunk értéket adni az $i_1, i_2, \ldots, i_k$ változóknak?

Nagyon egyszerűen! Ha ugyanis az $i_1, i_2, \ldots, i_k$ változók értékei a $0, 1, 2, \ldots, n-1$ tartományban mozognak, és így a változókat egy N természetes szám n-alapú számrendszerben felírt alakjának számjegyeinek képzeljük, akkor $0 \le N \le n^k - 1$. Vagyis programunkat például a következő módon írhatjuk meg:

```
Input n, k
For N := 0 to n^k − 1
    For t := 0 to k − 1
        i_t := N-nek t-edik számjegye n-alapú számrendszerben
    Next t
    ··· utasítások ···
Next N
```

Az N számokat n-alapú számrendszerben máris magunk előtt látjuk, amint számjegyeik az egymásba skatulyázott FOR-ciklusoknak megfelelően változtatják az i_t változó értékeit!

A fenti módszer egyetlen hátránya, hogy nagyon nagy N számokkal kell dolgoznunk, ráadásul nagy pontossággal! Például a Turbo Pascal *comp* típusú egész változóinak értéke legfeljebb $2^{63} \cong 9.22 \cdot 10^{18}$ lehet, ami például 10 egymásba skatulyázott ciklus esetén az i_t változók értékét $2^{6.3} \cong 78$-ban maximálja.

Persze a gyakorlat más. A programozásban valamelyest is járatosak az N szám n-alapú számrendszerbeli alakja számjegyeinek felderítése helyett inkább vesznek egy (legalább) k hosszú tömböt, s két ügyes kis ciklussal maguk változtatják a tömb elemeit, melyek az N szám jegyeit jelképezik. (Már magunk előtt is láthatjuk a ciklusváltozók, mint az n-alapú kilóméteróra számjegyeinek, gyors pörgését!)

2. Feladat. *Írjunk számítógép programot, amely tetszőleges sok egymásba ágyazott ciklus változóit számítja ki, és a változók tartománya is tetszőleges lehet!*

(Egy lehetséges megoldást a cikk végén ismertetünk.)

A cikkben bemutatott fenti két példa ötlete közös: *több kisebb számot (információt) kódoljunk egyetlen $N \in \mathbb{N}$ természetes számban!*

Olvasóink valószínűleg még sok más problémában alkalmazhatják a fenti ötletet, kíváncsian várjuk észrevételeiket!

Végezetül hadd álljon itt 8 éves Balázs fiam kérdése: „Kérdezni mindig könnyebb ..." jeligére:

3. Feladat* *Próbáljuk meg jellemezni az alábbi f_{mk} függvényeket: legyen $f_{mk}(n)$ az n szám m-alapú számrendszerben felírt alakját k-alapú számrendszerben kiolvasva kapott szám ($k \ge m$), azaz legyen*

$$f_{mk}(n) := \sum_{i=0}^{t} a_i k^i, \qquad ha \qquad n = \sum_{i=0}^{t} a_i m^i, \quad a_i \le m-1.$$

A 2. Feladat egy megoldása.

```
program szam2;

var k, n, t, l, ido: integer;
    ik : array [0,...,80] of integer;

write("Kérem a számrendszer alapszámát, azaz a ciklusváltozók tartományátÉ")
      readln(n);
write("és a számjegyek számát, azaz az egymásba ágyazott ciklusok számátÉ");
      readline(k);
for t := 0 to k - 1 do ik[t] := 0;
for l := 0 to k - 1 do write(ik[k - 1 - l]);            /* kezdő állapot */

while true do begin
if ik[0] > n - 1 then begin writeln("HIBA!); halt; end;
if ik[0] < n - 1 then ik[0] := ik[0] + 1
elseif ik[0] = n - 1 then begin
    l := 0;          /* az átfordulás helyének felderítése */
    while ik[l] > n - 2 do
          if ik[l] > n - 1 then halt
          else l := l + 1;
    if l > k then halt;
    if l = k then exit;
    if l < k then begin;
          ik[l] := ik[l] + 1;
          for t := 0 to l - 1 do ik[t] := 0;
          end; {if l}
    end; {else}
    goto xy(1,wherey);               /* kiírás */
for l. = 0 to k - 1 do write(ik[k - 1 - l] : 3);
end; {while}
```

IRODALOM

[1] Pintér Klára, *EGY ÖTLET: Lépjünk ki a tízes számrendszerből*, Polygon, VI. (1996), 77-85.

[2] Perelman, Ja. I., *Matematikai történetek és rejtvények*, Gondolat, Budapest, 1979.

dr. Szalkai István, Matematikai és Számítástechnikai Tanszék, Veszprémi Egyetem, 8200 Veszprém, Pf. 158.