**Problema 2 partit 90 de puncte**

**Autor prof. Szabó Zoltan – Liceul Tehnologic „Petru Maior”, Reghin**

**Inspectoratul Școlar Județean Mureș, Tg. Mureș**

**Descriere a unei/unor soluții posibile**

Să studiem numărul partițiilor ordonate pentru un număr natural *n*. Conform definiției ne interesează doar partițiile cu cel puțin doi termeni, însă studiul problemei se ușurează dacă acceptăm și partiția formată dintr-un singur termen.

Pentru *n*=1 avem o singură partiție:

1

Pentru *n*=2 avem două partiții:

1 1

2

Pentru *n=*3 avem 4 partiți:

1 1 1

1 2

2 1

3

Observăm că mulțimea partițiilor lui *n* poate avea ca prim element 1, 2, ..., n. Dacă o partiție începe cu 1, atunci celelalte elemente reprezintă o partiție pentru (*n*-1). Dacă primul element este 2, atunci celelalte elemente reprezintă o partiție pentru (*n-*2). Dacă primul element este *k*, atunci celelalte elemente reprezintă o partiție pentru (*n-k*).

Astfel, scriind toate partițiile lui *n* în ordine lexicografică obținem următorea listă:

1...

... toate partițiile lui (*n*-1) în ordine lexicografică

1..

2...

... toate partițiile lui (*n*-2) în ordine lexicografică

2...

...

*k ...*

*...* toate partițiile lui (*n-k*) în ordine lexicografică

*k ...*

*...*

*n-*1 1 (singura partiție a lui 1)

*n*  (singura partiție a lui 0)

Ne interesează numărul partițiilor lui *n*. Să notăm acest număr cu *P*(*n*). Avem următoarea formulă recursivă:

P(*n*) = P(*n*-1) + P(*n*-2) +...+ P(1) + P(0) , pentru orice *n*>1, și P(1)=P(0)=1

Aplicând formula pentru numărul *n*-1, obținem P(*n*-1) = P(*n*-2) +...+ P(1) + P(0)

Înlocuind în prima formulă, obținem P(*n*)=P(*n*-1)+P(*n*-1)=2\*P(*n-*1)

Dezvoltând formula mai departe, obținem: P(*n*) = 2\*P(*n-*1) = 2\*2\*P(*n-*2) = 2\*2\*2\*P(*n-*3)=...=2*n*-1P(1)

Avem nevoie de valorile acestui șir până la valoarea *n.* Elementele șirului sunt cuprinse în tabelul de mai jos, pentru valorile lui *n,* *n*≤15:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| P[i] | 1 | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |

Adică este adevărată formula P(*n*) = 2*n*-1, pentru orice *n*≥1,

În cazul în care ținem cont de faptul că partiția formată dintr-un singur termen nu se socotește, atunci din valoarea lui P(*n*) vom scădea 1, P(*n*)= 2*n*-1-1.

În enunțul problemei, pentru *n* =4 avem doar 7 termeni.

Ultima partiție este (3,1), iar partiția 4 – formată dintr-un singur termen – nu s-a mai introdus în listă. Acest termen despre care vorbim este ultimul în ordine lexicografică și nu afectează numerele de ordine pentru celelalte partiții.

Problema se poate rezolva cu programare dinamică folosind datele din tabelul de mai sus.

În continuare vom prezenta o variantă simplificată a algoritmului de mai sus, un algoritm greedy bazat pe observațiile și calculele anterioare.

Pentru a găsi cea de a *k* partiție a lui *n* în ordine lexicografică, vom ține cont de faptul că există 2*n*-1 partiții a lui *n*. Dintre acestea 2*n*-2 încep cu 1, iar celelalte 2*n*-2 partiții încep cu un număr mai mare decât 1.

Astfel, dacă *k*≤2*n*-2 atunci primul termen este 1 și trebuie să căutăm în continuare partiția lui (*n*-1) cu numărul de ordine *k*.

Dacă k>2*n*-2 permutarea va începe cu o valoare mai mare decât 1. Această valoare crește cumulativ cu 1 la fiecare pas când numărul ordinii lexicografice se găsește în a doua jumătate a listei, iar numărul de ordine căutat se micșorează cu o putere a lui 2, vom căuta într-o listă, din care vom elimina numerele care încep cu 1.

Datorită faptului că numărul de soluții 2*n*-1 depășește limitele tipului *unsigned long long* pentru n>63, trebuie să observăm încă o proprietate. Conform restricțiilor problemei, *n*<10000 , iar k are maximum 18 cifre. Am arătat mai sus, că dacă *k*≤2*n*-2, atunci primul termen al partiției este 1. Asta înseamnă, că pentru toate valorile lui *n* micșorate succesiv la n-1, n-2, n-3, ..., 64 șirul partițiilor se va completa cu 1, până când 2n-2 devine suficent de mic, să fie comparabil cu valoarea lui *k*, deci un număr reprezentabil pe 64 de biți.

Complexitatea algorimtului este O(n) (Dacă considerăm că numărul partițiilor lui *n* este egal cu M=2n-1, atunci algoritmul are complexitatea O(logM).

O soluție backtracking, care generează soluțiile în ordine lexicografică poate să obțină până la 54 de puncte, pentru testele în care valoarea lui *k* nu depășește 1000000.