



SG1

100 puncte

Fișiere sursă: **sg1.pas**, **sg1.cpp**, **sg1.c**

O misiune a echipei SG1 constă în activarea unui dispozitiv extraterestru acționat cu ajutorul unei tastaturi ciudate formate din **n** comutatoare aflate inițial toate în aceeași poziție (să o notăm cu 0). Se știe că trebuie setate (trecute în poziția 1) exact **k** comutatoare și că nu contează ordinea în care trebuie acționate comutatoarele. În plus, cu ajutorul unor documente antice, au aflat că între oricare două comutatoare succesive setate se pot afla cel puțin **d1** și cel mult **d2** comutatoare neselectate. De exemplu, pentru **n=7**, **k=3**, **d1=1** și **d2=2**, o configurație care corespunde cerinței este: **0100101**, în timp ce configurațiile **1010001**, **1100100**, **1010101** nu corespund datelor problemei.

Dacă o combinație de comutatoare setate nu activează dispozitivul, nu se întâmplă nimic deosebit (ce plictisitor episod!), ci comutatoarele se resetează automat, permițând încercarea altei combinații.

Se cere să se determine numărul maxim de configurații distincte de comutatoare setate pe care trebuie să le încerce echipa SG1 pentru a activa dispozitivul.

Cerință

Scrieți un program care, pentru valorile **n**, **k**, **d1**, **d2** date, determină numărul total de configurații posibile de comutatoare ce respectă condițiile din enunț.

Date de intrare

În fișierul text **sg1.in** se dau, pe aceeași linie, despărțite prin spații, valorile **n**, **k**, **d1** și **d2**.

Date de ieșire

În fișierul **sg1.out** se scrie numărul de configurații ce corespund cerinței.

Restricții:

$$0 < n < 101$$

$$0 < k \leq n$$

$$0 \leq d1 \leq d2 < n$$

Exemple:

sg1.in	sg1.out	Explicație
7 3 1 2	8	cele 8 configurații sunt: 1010100, 1010010, 1001010, 1001001, 0101010, 0101001, 0100101, 0010101
5 2 0 0	4	11000, 01100, 00110, 00011
14 8 1 5	0	

Timp maxim de rulare/test: 1.3 secunde sub Windows
0.6 secunde sub Linux