

Task 3. Rabbit

Pălărierul Nebun tocmai și-a pierdut iepurele preferat (Iepurele Alb, desigur) într-o secvență de N celule și încearcă să-l găsească. Celulele sunt numerotate cu numere întregi de la 1 la N . La început iepurele este situat într-o singură celulă necunoscută din secvență și fiecare secundă din căutarea Pălărierului decurge în felul următor:

1. La început alege o singură celulă din secvență și o verifică. O vom numi pe aceasta **celula verificată**. Dacă iepurele se află în aceasta, atunci căutarea se încheie.
2. După aceea, iepurele alege ori să rămână în aceeași celulă, ori să sară într-o celulă vecină (adică să sară ori o celulă în stânga ori o celulă în dreapta). **Rețineți că este posibil ca iepurele să sară în celula verificată, dacă aceasta este o celulă vecină; acest lucru nu înseamnă sfârșitul căutării!**

Deciziile iepurelui sunt deterministe în funcție de starea lui. Mai exact, iepurele are următoarele două stări:

1. **Speriat** - când iepurele se află în această stare, el **se îndepărtează de celula verificată**. Dacă nu se poate muta mai departe (adică el se află în celula 1 sau N), el rămâne în aceeași celulă.
2. **Curios** - când iepurele se află în această stare, el **se apropie de celula verificată**. Rețineți că întotdeauna este posibil ca iepurele să se apropie.

Rețineți că iepurele acționează numai conform ultimei celule verificate și nu ia în considerare celulele verificate anterior.

Din moment ce este iepurele preferat al Pălărierului, acesta știe exact starea iepurelui. Mai exact, Pălărierul știe că iepurele alternează între exact S secunde în care este *speriat* și C secunde în care este *curios*. De exemplu, dacă $S = 2$ și $C = 1$, starea iepurelui va fi secvența *[speriat, speriat, curios, speriat, speriat, curios...]*.

Pălărierul este foarte îngrijorat de iepurele său și vă roagă să scrieți un program `rabbit.cpp` care să găsească o secvență de celule de verificat, astfel încât este garantat că iepurele va fi găsit, indiferent de poziția inițială a acestuia.

Input

De pe prima linie a standard input-ului, programul trebuie să citească 3 numere întregi: N , S și C , reprezentând numărul de celule și comportamentul iepurelui.

Output

Pe prima linie a standard output-ului programul trebuie să afișeze K , numărul de secunde pe care le necesită secvența ta de căutare. Pe a doua linie, programul ar trebui să afișeze K întregi din intervalul $[1, N]$, enumerând celulele verificate în fiecare secundă. Rețineți că această secvență poate conține repetiții.

Scoring

Fie K numărul de secunde al căutării tale. Dacă încerci să verifici o celulă invalidă (adică în afara intervalului $[1, N]$) sau dacă secvența de verificări nu găsește **întotdeauna** iepurele, atunci programul va primi 0 puncte pentru acel test și verdictul *Wrong Answer*. Altfel, dacă punctajul unui test este de R puncte, vei primi pR puncte unde :

- $p = 0$, dacă $K > 2N$
- $p = 1$, dacă $K \leq T$
- $p = 0.3 \left(\frac{T}{K} \right)^2$, altfel

$$\text{unde: } T = \frac{N(S+C)}{S+2\max(S,C)} + 3\max(S,C)$$

Restricții

$$2 \leq N \leq 10^4, 0 \leq S, C \leq 50$$

Informații test

- Pentru 8% din teste $S=0, C=1$
- Pentru 12% din teste $S=1, C=0$
- Pentru 8% din teste $S=1, C=1$

Exemplu

Input	Output
12 2 1	14
	2 5 3 2 6 1 2 11 12 12 8 10 12 6

Explicație exemplu

Se poate verifica faptul că, indiferent de poziția inițială, această secvență va găsi întotdeauna iepurele. De exemplu, dacă vom considera că iepurele începe din celula 8, atunci căutarea se va desfășura în felul următor:

Secunda	Celula verificată	Starea iepurelui (Înainte de a se muta)	Mutarea iepurelui
1	2	Speriat	8 -> 9
2	5	Speriat	9 -> 10
3	3	Curios	10 -> 9
4	2	Speriat	10 -> 11
5	6	Speriat	11 -> 12
6	1	Curios	12 -> 11
7	2	Speriat	11 -> 12
8	11	Speriat	12 -> 12
9	12	Curios	Gasit

Pentru această soluție avem $K > T$, deoarece $K = 14$ și $T = 12$.

Astfel, procentajul de puncte primite pentru acest test este de $p \approx 0.26$.