Şablon — 100 puncte

Fişiere sursă: `sablon.cpp` sau `sablon.c` sau `sablon.pas`

Gigel şi Vasilică imaginează un mod de a transmite mesaje pe care nimeni să nu le poată descifra. Mesajul este ascuns într-un text care are `N` linii şi pe fiecare linie sunt exact `N` caractere – litere mari ale alfabetului englez, cifre, semne de punctuație și caracterul spațiu.
Decodificarea se face cu ajutorul unui şablon, de aceleaşi dimensiuni ca şi textul, care are câteva găuri. Suprapunând şablonul peste text rămân vizibile câteva caractere. Acestea se citesc în ordinea liniilor, de sus în jos, iar pe aceeaşi linie de la stânga la dreapta. Apoi hârtia cu textul se roteşte spre stânga, în sens trigonometric, cu 90°, şablonul rămânând fix. Alte caractere devin vizibile şi acestea se citesc în acelaşi mod. Operația se repetă de încă două ori (rotire cu 180°, respectiv cu 270°), până când textul ajunge, printr-o nouă rotație cu 90°, din nou în poziția inițială.
Din păcate, şablonul pentru codificare/decodificare s-a pierdut. În schimb a rămas la Gigel mesajul inițial iar la Vasilică a ajuns textul care conține mesajul.

Cerință
Să se reconstituie şablonul care a fost folosit la codificare.

Date de intrare
Fişierul de intrare **`sablon.in`** conține pe prima linie, mesajul inițial. Pe linia a doua a fişierului de intrare se găseşte valoarea `N`. Următoarele `N` linii conțin textul care ascunde mesajul.

Date de ieşire
Fişierul de ieşire **`sablon.out`** conține `N` linii a câte `N` caractere. Caracterele sunt '**`O`**' (pentru reprezentarea unei găuri) şi '**`X`**'.

Restricții
- prin rotirea textului nici una din găuri nu se va suprapune peste nici una din pozițiile ocupate de o gaură în pozițiile precedente ale textului
- $1 \le N \le 50$
- mesajul are maxim 1000 caractere şi se încheie cu un caracter diferit de spațiu
- în cazul în care există mai multe soluții, afişați una dintre ele

Exemplu

`sablon.in`	**`sablon.out`**
CODIFICARE CU SABLON 10 ABCD**C**EF**A**GH IJ**O**KL**E**MNOP **D**QRSTUVW**C**X YZA**I**BC**R**DEF G**F**HIJKLMN**I** **A**JKLMNOP**S**Q RST**O**UV WXY Z**B**ABCDEFG**U** HIJK**N**LM**C**NO PQ**L**RS TUVW	XXXX**O**XXXXX XX**O**XXXXXXX **O**XXXXXXXXX XXX**O**XXXXXX X**O**XXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX

Timp maxim de execuție/test: 1 secundă

Şir — 100 puncte

Fişiere sursă: `sir.cpp` sau `sir.c` sau `sir.pas`

Gigel se distrează construind şiruri crescătoare de numere din mulțimea {1,2,…,n}. La un moment dat observă că unele şiruri, de cel puțin **k** termeni (**k**≥3), au o proprietate mai aparte: diferența dintre doi termeni consecutivi este constantă. Iată câteva exemple de astfel de şiruri pentru n≥22:

2,3,4
1,5,9,13
7,10,13,16,19,22

Cerință

Dându-se numărul natural **n** ajutați-l pe Gigel să numere câte astfel de şiruri poate să construiască.

Date de intrare

În fişierul de intrare **`sir.in`** se găseşte, pe prima linie, numărul n.

Date de ieşire

În fişierul de ieşire **`sir.out`** se va afişa, pe prima linie, numărul cerut urmat de caracterul sfârşit de linie.

Restricții:

- 3 ≤ n ≤ 20000
- 3 ≤ k ≤ n

Exemple:

sir.in	**sir.out**
3	1
4	3
5	7

Timp maxim de execuție/test: 1 secundă.

Snipers — 100 puncte

Fişiere sursă: `snipers.pas` sau `snipers.c` sau `snipers.cpp`

Se spune că în timpul războiului cu gnomii, trolii au trimis **n** trăgători de elită să lichideze cele **n** căpetenii inamice.

Din fericire căpeteniile inamice erau plasate în câmp deschis, iar trăgătorii au reuşit să se plaseze în zonă fără să fie observați. Când să fie dată comanda de tragere s-a constatat că nu se transmisese fiecărui trăgător ce căpetenie să împuşte, iar dacă doi trăgători ar fi tras în aceeaşi căpetenie sau traiectoriile razelor ucigaşe s-ar fi intersectat, atunci ar fi scăpat cel puțin o căpetenie care ar fi putut duce războiul până la capăt, iar trolii ar fi fost învinşi. Deoarece căpeteniile aveau capacitatea de a deveni invizibile oricând doreau (pe o perioadă nelimitată), trebuiau lichidate simultan, altfel… Istoria ne spune că trolii au învins deoarece comandantul lor a reuşi ca în mai puțin de o secundă să transmită fiecărui trăgător în ce căpetenie să tragă. Voi puteți face asta?

Cerință

Scrieți un program care citind pozițiile trăgătorilor şi a căpeteniilor determină căpetenia în care trebuie să tragă fiecare trăgător.

Date de intrare

Fişierul de intrare **`snipers.in`** conține pe prima sa linie numărul **n.** Pe următoarele **n** linii se află perechi de numere întregi, separate prin spațiu, ce reprezintă coordonatele trăgătorilor urmate de alte **n** perechi de numere întregi ce reprezintă coordonatele căpeteniilor(abscisă şi ordonată).

Date de ieşire

Fişierul de ieşire **`snipers.out`** conține **n** linii. Pe linia *i* a fişierului se află numărul căpeteniei țintite de trăgătorul *i (i=1..n).*

Restricții

- `0 < n < 200`
- Coordonatele sunt numere întregi din intervalul [0, 50000]
- Raza ucigaşă a oricărei arme se opreşte în ținta sa.
- În datele de intrare nu vor exista trei persoane aflate în puncte coliniare.

Exemplu

snipers.in	**snipers.out**	**snipers.in**	**snipers.out**
2 1 3 1 1 3 4 3 1	1 2	5 6 6 4 13 2 8 9 4 5 2 6 11 9 7 3 9 1 4 7 3	2 1 3 4 5

Timp maxim de execuție/test: 1 secundă