

## Zadaci

---

Zadatak	TO	PROZORI	AVION	LAKAT
Izvršna datoteka	to.exe	prozori.exe	avion.exe	lakat.exe
Izvorni kôd	to.pas to.c to.cpp	prozori.pas prozori.c prozori.cpp	avion.pas avion.c avion.cpp	lakat.pas lakat.c lakat.cpp
Ulazna datoteka	to.in	prozori.in	avion.in	lakat.in
Izlazna datoteka	to.out	prozori.out	avion.out	lakat.out
Vremensko ograničenje (po test podatku)	10 sekundi	10 sekundi	10 sekundi	10 sekundi
Broj test podataka	10	10	10	10
Broj bodova (po test podatku)	3	4	6	7
Ukupno bodova	30	40	60	70
	200			

## TO

---

Mirko i Slavko igraju jednu jako zanimljivu igru.

Ploča za tu igru nema neko prikladno ime pa ju za ovu priliku nazovimo '**to**'. 'To' stoji uspravno i sastoji se od 6 redova i 7 stupaca. Igra se tako da igrači naizmjenično ubacuju žetone u 'to'. Prilikom ubacivanja žeton pada kroz stupac i ostaje u najnižem redu u koji može pasti tj. red iznad najvišeg žetona u stupcu ili u najdonjem redu ako je stupac prazan. Svaki žeton zauzima točno jedno polje.

Svaki od igrača ima na raspolaganju 21 žeton i **Mirko**, jer slabije igra, uvijek igra **prvi**. Mirko ima žute žetone, a Slavko crvene.

Cilj igre je spojiti **četiri uzastopna** žetona iste boje vodoravno, okomito ili dijagonalno. Situacija u kojoj se to dogodi zove se pobjednička situacija.

Igrajući tu igru, Mirko i Slavko ne vode računa o tome je li igra (i kada je) gotova jer bi im to bilo prenaporno pa oni igraju sve dok ne ubace sve žetone. Stoga su odlučili zapisivati tko je u koji stupac ubacio žeton i prepustiti nekom pametnijem da ih na osnovu zapisa obavijesti o pobjedniku.

Napišite program koji će odrediti **tko** je pobijedio i u **kojem** potezu. U slučaju da se tijekom igre više puta dogodila pobjednička situacija, ona koja se dogodila **prva određuje** pobjednika. Također, postoji mogućnost da je igra završena neriješeno tj. da nitko nije pobijedio.

### Ulazni podaci

Ulazna datoteka sastoji se od 21 retka.

U i-tom retku se nalaze dva cijela broja  $M_i$  i  $S_i$ , međusobno odvojena jednim razmakom. Broj  $M_i$  označava broj stupca u kojeg Mirko ubacuje žeton u i-tom potezu, a broj  $S_i$  broj stupca u kojeg Slavko ubacuje žeton u i-tom potezu.

Stupci su označeni brojevima od 1 do N.

### Izlazni podaci

U prvi i i jedini redak izlazne datoteke treba zapisati '**mirko P**' ili '**slavko P**' ako je Mirko odnosno Slavko pobijedio u P-tom potezu ili '**remi**' ako je igra završena bez pobjednika.

# TO

---

## Test primjeri

**to.in**

2 2  
5 2  
3 7  
6 1  
4 6  
3 1  
3 5  
3 3  
6 3  
2 5  
4 1  
6 2  
2 5  
7 5  
1 7  
4 4  
4 1  
7 6  
1 7  
7 5  
6 4

**to.out**

mirko 5

**to.in**

1 1  
2 4  
6 4  
5 1  
1 2  
7 4  
3 4  
7 7  
4 6  
3 5  
2 4  
6 1  
7 3  
6 3  
6 1  
6 3  
2 5  
7 3  
2 7  
2 5  
5 5

**to.out**

slavko 7

**to.in**

1 2  
1 2  
1 2  
2 1  
2 1  
2 1  
3 4  
3 4  
3 4  
4 3  
4 3  
4 3  
5 6  
5 6  
5 6  
5 6  
6 5  
6 5  
6 5  
6 5  
7 7  
7 7  
7 7

**to.out**

remi

# PROZORI

---

U popularnom i izrazito stabilnom operativnom sustavu kojeg svi silom prilika koristimo otvorili smo hrpu prozora.

Svaki prozor je pravokutnik koji se sastoji od jediničnih kvadratića (dimenzija 1 x 1).

Novootvoreni prozor, ovisno o svojoj poziciji i veličini, **može** (djelomično ili potpuno) prekriti neke prethodno otvorene prozore.

Prozor zatvaramo tako da mišem kliknemo na njegov **gornji desni** kvadratić koji u tom trenutku **mora biti vidljiv**. Kvadratić nekog prozora je vidljiv ako ga ne prekriva nijedan naknadno otvoreni prozor koji nije već zatvoren.

Napišite program koji će izračunati **minimalni** broj klikova da **zatvorimo prvootvoreni** prozor.

## Ulazni podaci

U prvom retku ulazne datoteke nalazi se cijeli broj  $N$ , broj otvorenih prozora,  $1 \leq N \leq 100$ .

U svakom od sljedećih  $N$  redaka nalaze se četiri cijela broja  $R1$ ,  $S1$ ,  $R2$  i  $S2$ , međusobno odvojena s po jednim razmakom,  $1 \leq R1 \leq R2 \leq 10000$ ,  $1 \leq S1 \leq S2 \leq 10000$ . Brojevi  $R1$  i  $S1$  označavaju redak i stupac ekrana u kojem se nalazi **gornji lijevi kvadratić** prozora, dok  $R2$  i  $S2$  označavaju redak i stupac ekrana u kojem se nalazi **donji desni kvadratić** prozora. Prozori su otvarani **istim** redoslijedom kojim su zapisani u ulaznoj datoteci.

Ekran zamišljamo da je organiziran u retke i stupce jediničnih kvadratića, pri čemu su reci numerirani odozgo na dolje, stupci slijeva na desno, a gornji lijevi kvadratić na ekranu se nalazi u prvom retku i prvom stupcu.

## Izlazni podaci

U prvi i jedini redak izlazne datoteke treba zapisati traženi minimalni broj klikova.

## Test primjeri

**prozori.in**

```
3
3 1 6 4
1 2 4 6
2 3 5 5
```

**prozori.out**

```
3
```

**prozori.in**

```
3
4 1 6 3
2 2 5 5
1 4 3 6
```

**prozori.out**

```
3
```

**prozori.in**

```
3
3 3 4 4
1 1 2 2
5 5 6 6
```

**prozori.out**

```
1
```

# AVION

---

Zamislimo avion u kojem postoji samo jedan uski prolaz kroz redove sjedala i putnike koji ulaze i sjedaju na svoja mjesta.

Svaki putnik mora sjesti točno na svoje mjesto, a ulaz u avion se nalazi na početku aviona, neposredno ispred prvog reda. Putnici ulaze jedan iza drugog **bez** nepotrebnog zaustavljanja i odugovlačenja.

Putnik se hodajući kroz prolaz prema svom mjestu u svakom redu zadržava **točno jednu** sekundu (**ili više** ako je ispred njega neki zastoj) sve dok ne stigne do svog reda i tada se u prolazu zadržava **točno pet** sekundi radi spremanja stvari u prostor iznad sjedala. U svakoj sekundi u prolazu u ravnini nekog reda može se nalaziti **samo jedan** putnik.

Napišite program koji će izračunati koliko je vremena potrebno da prolaz postane **slobodan** tj. da svi putnici sjednu na svoja mjesta.

## Ulazni podaci

U prvom retku ulazne datoteke nalazi se cijeli broj  $N$ , broj putnika,  $1 \leq N \leq 1000$ .

U  $(i+1)$ -om retku nalazi se cijeli broj  $R_i$ , broj reda u koji treba sjesti  $i$ -ti putnik,  $1 \leq R_i \leq 1000$ .

Putnici su označeni brojevima od 1 do  $N$  i ulaze tim redoslijedom, a broj putnika koji sjede u istom redu **nije ograničen**.

## Izlazni podaci

U prvi i jedini redak izlazne datoteke treba zapisati vrijeme (u sekundama) potrebno da se svi putnici smjeste.

## Test primjeri

**avion.in**

1  
3

**avion.out**

7

**avion.in**

2  
3  
3

**avion.out**

12

**avion.in**

4  
4  
4  
1  
5

**avion.out**

19

# LAKAT

---

Mirko i Slavko igraju jednu jako zanimljivu igru.

Svako od njih zamisli jednu riječ, zatim sjednu za računalo i istovremeno se naguravajući i udarajući jedan drugog **laktom** u bubrege utipkavaju svaki svoju riječ.

Nakon što su utipkali riječi pogledaju u monitor i imaju što i vidjeti. Njihove riječi su tako **isprepletene** da ne mogu odgonetnuti koje slovo pripada čijoj riječi, a oni bi to baš tako jako htjeli saznati.

Napišite program koji će **za svako** slovo iz riječi na monitoru **odrediti** da li pripada Mirkovoj ili Slavkovoj riječi.

**Napomena:** rješenje ne mora biti jedinstveno.

## Ulazni podaci

U prvom i drugom retku ulazne datoteke nalaze se Mirkova i Slavkova riječ. Svaka od te dvije riječi se sastoji samo od malih slova engleske abecede (a-z), a broj znakova u svakoj od riječi je manji ili jednak od 150.

U trećem retku nalazi se riječ koju su Mirko i Slavko ugledali na monitoru, nastala ispreplitanjem prve dvije riječi.

Ulazni podaci će biti takvi da će **uvijek** postojati rješenje.

## Izlazni podaci

U prvi i jedini redak izlazne datoteke treba **za svako** slovo iz treće riječi zapisati broj '1' ili broj '2' tako da se čitanjem slova iz treće riječi koja se nalaze na pozicijama označenima brojem '1' dobije **prva** riječ, a čitanjem slova iz treće riječi koja se nalaze na pozicijama označenima brojem '2' dobije **druga** riječ.

## Test primjeri

**lakat.in**

novine  
vesna  
novesvinena

**lakat.out**

11222111122

**lakat.in**

tata  
mama  
mtatamaa

**lakat.out**

21112212

**lakat.in**

hsin  
sinh  
hsinhsin

**lakat.out**

12222111