

ZADATAK	JOKER	FLASH	PITON
izvorni kôd	joker.pas joker.c joker.cpp joker.cxx joker.py	flash.pas flash.c flash.cpp flash.cxx flash.py	python.pas python.c python.cpp python.cxx python.py
izvršna datoteka	joker.exe	flash.exe	python.exe
ulazni podaci	standardni ulaz		
izlazni podaci	standardni izlaz		
vremensko ograničenje	1 sekunda		
broj bodova	50	70	80
	200		



Agencija za odgoj i obrazovanje  
Education and Teacher Training Agency



HRVATSKI SAVEZ  
INFORMATIČARA



Ministarstvo znanosti,  
obrazovanja i sporta

Popularna igra na sreću sastoji se od izvlačenja 6 kuglica iz bubnja na kojima su napisani različiti prirodni brojevi između 1 i 45. Dobitni *joker broj* se formira iz izvučenih brojeva tako da se istim redoslijedom kojim su kuglice izvučene zapišu njihove zadnje znamenke. Na primjer, ako su redom izvučeni brojevi 12, 35, 1, 2, 23 i 39, onda je joker broj 251239.

Igrači kupuju dobitne listiće te s nestrpljenjem čekaju izvlačenje i proračun joker broja kako bi izračunali svoj mogući dobitak. Dobitak listića računa se tako što se serijski broj napisan na listiću uspoređi sa izračunatim joker brojem na način da se prebroji koliko se zadnjih znamenki ova dva broja poklapa, kao što je prikazano na sljedećoj tablici.

Serijski brojevi	Naziv dobitka
251239	I. vrsta
X51239	II. vrsta
XX1239	III. vrsta
XXX239	IV. vrsta
XXXX39	V. vrsta
XXXXXX	Nedobitan

U lijevom stupcu se nalaze serijski brojevi listića, gdje *X* označava proizvoljne znamenke. U desnom stupcu tablice nalaze se nazivi dobitaka za svaki od oblika serijskih brojeva. Dakle, ako je serijski broj točno jednak Joker broju onda dobivamo dobitak I. vrste, ako je zadnjih 5 znamenaka jednako dobivamo dobitak II. vrste, i tako dalje, sve do dobitka V. vrste koji dobivamo kada su zadnje dvije znamenke jednake. Svi ostali serijski nisu dobitni.

Napišite program koji će za zadane izvučene brojeve i tri serijska broja listića **ispisati nazive dobitaka** za svaki od njih. Nazive dobitaka je potrebno ispisati točno onako kako piše u gornjoj tablici (broj dobitka se piše rimskim brojevima pomoću velikih slova 'I' i 'V', nakon broja slijedi točka i razmak te malim slovima 'vrsta').

### **ULAZNI PODACI**

U prvom redu nalazi se točno šest različitih prirodnih brojeva manjih ili jednakih 45 koji predstavljaju brojeve na kuglicama izvučenim iz bubnja.

U svakom od sljedeća tri reda ulaza nalazi se po jedan niz od šest znamenaka koji predstavlja serijski broj listića. Primijetite da serijski broj uvijek ima šest znamenaka te može imati vodeće nule.

### **IZLAZNI PODACI**

U svaki od tri reda potrebno je ispisati naziv dobitka za odgovarajući serijski broj.

**PRIMJERI TEST PODATAKA**

<b>ulaz</b>  12 35 1 2 23 39 151239 251229 251339	<b>ulaz</b>  5 45 35 25 15 1 555551 235551 555552	<b>ulaz</b>  2 17 33 12 39 44 000022 001194 232294
<b>izlaz</b>  II. vrsta Nedobitan V. vrsta	<b>izlaz</b>  I. vrsta III. vrsta Nedobitan	<b>izlaz</b>  Nedobitan V. vrsta IV. vrsta

Flash memorija je vrsta elektroničke memorije koja ne gubi informacije kada se prekine napon. Mirko je nedavno patentirao i proizveo novu vrstu jeftine memorije koju je nazvao MOR. MOR je naročito jeftina u proizvodnji, ali je složenija za rukovanje zbog svojih određenih ograničenja.

MOR memorija se sastoji od niza od  $M$  blokova, gdje se svaki blok sastoji od točno  $K$  bitova. Kod MOR memorije nije uvijek moguće postaviti pojedini bit na željenu vrijednost već su dopuštene samo sljedeće operacije:

- Pojedini bit možemo postaviti na 0 te ova operacija traje 1 milisekundu.
- Sve bitove u pojedinom bloku možemo postaviti na 1 te ova operacija traje 100 milisekundi.

Napišite program koji će za zadano početno i traženo stanje memorije pronaći najmanje vrijeme potrebno da se memorija iz početnog stanja dovede u traženo.

### ULAZNI PODACI

U prvom redu nalaze se dva prirodna broja,  $M$  i  $K$  ( $M, K \leq 20, M \cdot K \leq 80$ ) međusobno odvojena razmakom - broj blokova i broj bitova u pojedinom bloku.

U drugom redu nalazi se niz od točno  $M \cdot (K+1) - 1$  znakova - početno stanje memorije. U trećem redu nalazi se niz od točno  $M \cdot (K+1) - 1$  znakova - traženo stanje memorije.

Početno i traženo stanje memorije su nizovi znakova koji se sastoje od  $M$  blokova međusobno odvojenih znakom '|' (vertikalna crta, ASCII 124), a svaki blok se sastoji od točno  $K$  znakova '0' ili '1' koje predstavljaju vrijednost određenog bita u bloku.

### IZLAZNI PODACI

U prvi i jedini red izlaza potrebno je ispisati najmanje moguće vrijeme u milisekundama potrebno da se memorija postavi u traženo stanje.

### PRIMJERI TEST PODATAKA

<b>ulaz</b> 2 4 0110 1000 0000 0000	<b>ulaz</b> 2 4 0110 1000 0000 0001	<b>ulaz</b> 3 3 110 011 111 101 111 011
<b>izlaz</b> 3	<b>izlaz</b> 105	<b>izlaz</b> 202

Piton je strukturirani programski jezik sintaksom donekle sličan popularnom Pythonu. Ovaj jednostavni jezik ima samo dvije naredbe:

- $Px$  gdje je  $x$  malo slovo engleske abecede: ova naredba ispisuje slovo  $x$ .
- $Fk$  gdje je  $k$  prirodni broj manji ili jednak od 20 zapisan bez vodećih nula: ova naredba ponavlja točno  $k$  puta blok naredbi koji slijedi.

Blokovi naredbi se specificiraju uvlačenjem naredbi, ali se koristi znak '.' (točka) umjesto uobičajenih razmaka. Točnije, blokovi i programi u Pitonu se definiraju na sljedeći način: Za nenegativni cijeli broj  $N$  definiramo  $N$ -*uvučeni blok* kao niz linija koji se sastoji od jedne ili više:

- Naredbe  $P$  uvučene s  $N$  točaka.
- Naredbe  $F$  uvučene s  $N$  točaka nakon koje slijedi točno jedan  $M$ -uvučeni blok gdje je  $M$  cijeli broj veći od  $N$ .

Jedan blok može sadržavati proizvoljan broj  $P$  i  $F$  naredbi proizvoljnim redosljedom. Kao što je gore definirano, nakon svake  $F$  naredbe slijedi blok koji mora biti više uvučen nego sama naredba, ti blokovi također mogu sadržavati druge blokove itd. *Program* u Pitonu je jedan 0-uvučeni blok (dakle blok koji nije uopće uvučen pa prva linija ne sadrži točku).

U sljedećoj tablici dan je jedan program u Pitonu te njemu ekvivalentni programi u C++-u i u Pascal-u.

<pre>F3 ..Pa ..Pb ..F4 ...Pc ..F2 .....Pd .....Pe ..Pa</pre>	<pre>for (int i=0; i&lt;3; i++) {     printf("a");     printf("b");     for (int j=0; j&lt;4; j++) {         printf("c");     }     for (int j=0; j&lt;2; j++) {         printf("d");         printf("e");     }     printf("a"); }</pre>	<pre>FOR i := 1 TO 3 DO BEGIN     write('a');     write('b');     FOR j := 1 TO 4 DO BEGIN         write('c');     END;     FOR j := 1 TO 2 DO BEGIN         write('d');         write('e');     END;     write('a'); END;</pre>
--	---	--

Napišite program koji će, za zadani program u Pitonu, pronaći **koliko kojih malih slova** zadani program ispisuje.

### ULAZNI PODACI

U prvom redu nalazi se prirodni broj,  $L$  ( $L \leq 100$ ) - broj linija u zadanom programu. Svaki od sljedećih  $L$  redova sadrži niz od najviše 80 znakova - jednu liniju zadanog programa. Jedini znakovi koji se pojavljuju su točke, mala slova engleske abecede, velika slova 'F' i 'P' te znamenke '0' do '9'. Ulaz je ispravan program u Pitonu prema pravilima iz teksta zadatka koji ispisuje ukupno najviše 10000 znakova.

### IZLAZNI PODACI

Potrebno je ispisati koliko se kojih slova pojavljuje u izlazu zadanog programa. Točnije, potrebno je ispisati onoliko redova koliko se različitih malih slova pojavljuje u izlazu zadanog programa. Za svako slovo  $x$  koje se pojavljuje u izlazu potrebno je ispisati jedan red oblika ' $x k$ ', gdje je  $k$  ukupan broj pojavljivanja tog slova u izlazu. Slova trebaju biti **ispisana abecednim redosljedom**.

**PRIMJERI TEST PODATAKA**

<pre>ulaz  5 Px Py F3 ..Px ..Pz  izlaz  x 4 y 1 z 3</pre>	<pre>ulaz  9 F3 ..Pa ..Pb ..F4 ...Pc ..F2 .....Pd .....Pe ..Pa  izlaz  a 6 b 3 c 12 d 6 e 6</pre>	<pre>ulaz  8 Pa F3 .F12 ..Px ..F20 ...F2 ....F3 .....Pa  izlaz  a 4321 x 36</pre>
---	---	---