



KANTON SARAJEVO

MINISTARSTVO ZA ODGOJ I OBRAZOVANJE  
KANTONA SARAJEVO

---

# Kantonalno takmičenje iz informatike za srednje škole

---

Bilten pripremili:

Hadžić Hadžem

Zavlan Amer

Džaka Tarik

Parić Muamer

16. Mart 2024. GODINE

<b>Rezultati</b>	<b>3</b>
Poredak takmičara	3
Poredak škola	5
<b>Postavke zadataka</b>	<b>6</b>
Zadatak 1 - Portal	6
Zadatak 2 - Tastatura	8
Zadatak 3 - Eksplozija	10
Zadatak 4 - Univerziteti	12
Zadatak 5 - Skakač	15
<b>Rješenja zadataka</b>	<b>18</b>
Zadatak 1 - Portal	18
Zadatak 2 - Tastatura	19
Zadatak 3 - Eksplozija	20
Zadatak 4 - Univerzitet	22
Zadatak 5 - Skakač	24



# Rezultati

## Poredak takmičara

Mjesto	Ime i prezime	Škola	Z1	Z2	Z3	Z4	Z5	Ukupno
1	Benjamin Mujkić	JU Druga gimnazija	100	100	100	100	70	470
2	Faruk Ibrahimović	JU Druga gimnazija	100	100	96.875	100	40	436.875
3	Farah Demirović	Richmond Park International Secondary School	100	100	100	100	0	400
4	Alen Avdibegović	JU Druga gimnazija	100	100	100	5	60	365
5	Mirza Bušatlić	JU Druga gimnazija	100	100	100	0	60	360
6	Faruk Demirović	Richmond Park International Secondary School	100	100	100	10	0	310
7	Nadir Hrustanbegović	Treća gimnazija Sarajevo	100	100	90.625	0	10	300.625
8	Nidal Vatreš	Richmond Park International Secondary School	100	100	100	0	0	300
9	Amer Bečarević	JU Četvrta gimnazija Ilidža	100	100	100	0	0	300
10	Faris Hrustić	Treća gimnazija Sarajevo	100	100	100	0	0	300
11	Adnan Osmić	JU Druga gimnazija	100	100	62.5	35	0	297.5
12	Appa Bugis Mubarak	Srednja elektrotehnička škola Sarajevo	100	100	81.25	0	0	281.25
13	Beriz Dautović	Srednja elektrotehnička škola Sarajevo	100	100	75	0	0	275
14	Abddullah Ferhatbegović	Richmond Park International Secondary School	100	100	62.5	0	0	262.5
15	Tarik Dacić	JU Druga gimnazija	100	100	53.125	0	0	253.125
16	Avdić Muhamed	JU Četvrta gimnazija Ilidža	100	100	40.625	0	0	240.625



17	Faruk Hodžić	JU Prva gimnazija Sarajevo	100	100	31.25	0	0	231.25
18	Emir Salčinović	Treća gimnazija Sarajevo	100	100	31.25	0	0	231.25
19	Galijašević Damir	JU Gimnazija Obala Sarajevo	100	100	0	0	0	200
20	Tarik Smajović	Treća gimnazija Sarajevo	100	0	78.125	0	0	178.125
21	Latif Abazi	Prva bošnjačka gimnazija Sarajevo	100	18	6.25	0	0	124.25
22	Imran Šeta	Prva bošnjačka gimnazija Sarajevo	100	18	3.125	0	0	121.125
23	Fadil Masnopita	Srednjoškolski centar "Nedžad Ibrišimović" Ilijaš	100	18	0	0	0	118
24	Žiga Mak	JU Gimnazija Obala Sarajevo	100	18	0	0	0	118
25	Adna Mujić	JU Prva gimnazija Sarajevo	100	18	0	0	0	118
26	Anel Kadrić	Srednjoškolski centar "Nedžad Ibrišimović" Ilijaš	100	18	0	0	0	118
27	Imad Husić	JU Četvrta gimnazija Ilidža	100	0	0	0	0	100
28	Ahmed Hadžiomerović	JU Gimnazija Dobrinja	100	0	0	0	0	100
29	Danin Sadžak	JU Prva gimnazija Sarajevo	100	0	0	0	0	100
30	Hamza Veljan	Srednja elektrotehnička škola Sarajevo	100	0	0	0	0	100
31	Filip Balaban	JU Gimnazija Dobrinja	67	0	0	0	0	67
32	Rijad Karadža	JU Gimnazija Dobrinja	67	0	0	0	0	67
33	Anur Berbić	Srednja škola metalnih zanimanja Sarajevo	0	0	0	0	0	0
34	Tarik Egrlić	Srednja škola metalnih zanimanja Sarajevo	0	0	0	0	0	0
35	Harun Seljubac	Prva bošnjačka gimnazija Sarajevo	0	0	0	0	0	0
36	Edis Mušanović	Srednjoškolski centar "Nedžad Ibrišimović" Ilijaš	0	0	0	0	0	0
37								
38								



## Poredak škola

Redno mjesto	Naziv škole	Ukupno bodova
1	JU Druga gimnazija	1221.875
2	Richmond Park International Secondary School	1006.875
3	Treća gimnazija Sarajevo	822.5
4	Srednja elektrotehnička škola Sarajevo	656.25
5	JU Četvrta gimnazija Ilidža	637.5
6	JU Prva gimnazija Sarajevo	449.25
7	JU Gimnazija Obala Sarajevo	318
8	Prva bošnjačka gimnazija Sarajevo	245.375
9	Srednjoškolski centar "Nedžad Ibrišimović" Ilijaš	236
10	JU Gimnazija Dobrinja	234
11	Srednja škola metaliskih zanimanja Sarajevo	0



# Postavke zadatka

## Zadatak 1 - Portal

---

Duboko u šumama Čvrsnice nalazi se drevni portal poznat kao “Portal parnosti”. Legenda kaže da ovaj portal povezuje dva paralelna svijeta - jedan u kojem vladaju parni brojevi i drugi u kojem vladaju neparni brojevi. Vaš zadatak, neustrašivi programeru, je da dešifirate kriptični uzorak koji otključava ovaj portal. Samo razumijevanjem izmjenjujućeg plesa parnosti i neparnosti možete se nadati da ćete preći njegov sjajni prag.

Zadatak se sastoji od određivanja da li se na parnim mjestima datog niza nalaze samo parni brojevi, te na neparnim mjestima nalaze samo neparni brojevi. Ukoliko niz poštuje ovaj obrazac potrebno je ispisati “Portal se otvara”, a u suprotnom potrebno je ispisati “Portal ostaje zatvoren”. Kao pomoć u rješavanju ovog zadatka, prisjećate se drevnih spisa za provjeru parnosti broja u C++ programskom jeziku:

```
if(A % 2 == 0) cout << "Broj A je paran";  
else cout << "Broj A je neparan";
```

### Ulazni podaci

Prvi red ulaza se sastoji od  $n$ , dužine niza.

Drugi i posljednji red ulaza se sastoji od  $n$  brojeva  $p_i$ , brojeva niza iz opisa zadatka.

### Ograničenja

$$1 \leq n \leq 100,$$

$$0 \leq p_i \leq 99$$

Podzadatak 1 (33 boda)

Niz sadrži samo brojeve 0 i 1.



Podzadatak 2 (34 boda)

Niz sadrži tačno 4 broja, odnosno  $n = 4$ .

Podzadatak 3 (33 boda)

Bez dodatnih ograničenja.

### Izlazni podaci

Izlaz treba sadržavati tekst “Portal se otvara” ako se na neparnim mjestima niza (prvi broj, treći broj, ...) nalaze samo neparni brojevi, a na parnim mjestima (drugi broj, četvrti broj, ...) nalaze samo parni brojevi. U suprotnom potrebno je ispisati “Portal ostaje zatvoren”.

### Primjer

Ulaz	Očekivani izlaz	Objašnjenje
3 1 4 4	Portal ostaje zatvoren	Na trećem (neparnom) mjestu u nizu se nalazi paran broj, tako da uslov nije ispunjen.
7 1 0 1 0 1 0 1	Portal se otvara	0 je paran broj, 1 je neparan broj, tako da je ispunjen uslov da se na parnim mjestima nalaze parni brojevi, a na neparnim mjestima neparni brojevi.
4 30 42 55 21	Portal ostaje zatvoren	Prvi broj je paran, tako da uslov nije ispunjen.



## Zadatak 2 - Tastatura

---

Nakon što se vratio sa olimpijskih igara, Kemica je otkrio da mu je brat Ramiz zamijenio tipke na tastaturi. To je učinio tako što je  $N$  puta odabrao dvije tipke  $T_1$  i  $T_2$ , te zamijenio njihove pozicije. Kemica se nije naljutio, već je odlučio da se našali sa svojim bratom na način da se pretvara da mu to ne smeta. Potrebno je da mu pomognete tako što ćete mu reći koja će riječ biti ispisana ukoliko unese zadanu riječ. Kemica zna koje tipke je Ramiz zamijenio i u kojem redoslijedu.

Ako dvije tipke A i B zamijene mjesta, to znači da će pritiskom na mjesto gdje sada stoji tipka A u računar biti otkucano slovo 'B', dok će pritiskom na mjesto gdje sada stoji tipka B biti otkucano slovo 'A'.

### Ulazni podaci

Prvi red sadrži jedan broj  $N$ , broj zamjena koje je Ramiz napravio. Narednih  $N$  redova sadrže po dva mala slova engleske abecede,  $T_{i1}$  i  $T_{i2}$ , razdvojena razmakom. Ovo označava dvije tipke koje je Ramiz zamijenio. Konačno, posljednji red sadrži riječ dužine  $D$  koja također sadrži samo mala slova engleske abecede - riječ koju je Kemica pokušao otkucati.

### Ograničenja

$$1 \leq N \leq 30,$$

$$0 \leq D \leq 100$$

Podzadatak 1 (18 boda)

Ramiz je zamijenio samo jedan par tipki na tastaturi, odnosno  $N = 1$ .

Podzadatak 2 (35 boda)

Ramiz je mijenjao samo tipke sa slovima A, B, C i D.

Podzadatak 3 (47 boda)





Bez dodatnih ograničenja.

## Izlazni podaci

Izlaz se sastoji od jedne riječi, koja predstavlja riječ koja će biti ispisana ukoliko Kemica pokuša unijeti riječ zadanu na ulazu.

## Primjer

Ulaz	Očekivani izlaz	Objašnjenje
2 k t p a kemica	temicp	Ako Kemica pritisne slovo "t" otkucat će se slovo "k". Slova "e", "m", "i" i "c" su ostala nepromijenjena. Konačno, kako bi otkucao slovo "a", mora pritisnuti tipu sa slovo "p".
4 a b a c b c a c abcd	abcd	Sve tipke su se vratile na svoje mjesto. Originalno: a b c Prva izmjena: b a c Druga izmjena: b c a Treća izmjena: c b a Četvrta izmjena: a b c
5 v e k u e k w e t m vucko	ekcvo	



## Zadatak 3 - Eksplozija

---

Mario se voli igrati sa dvije stvari: vatrom i nizom znakova. Niz znakova je karakterističan zbog toga što se u njemu nalazi i kraći niz koji predstavlja “eksploziju”. Ukoliko Mario pride vatri preblizu, pokreće se lančana reakcija.

Lančana reakcija se odvija na sljedeći način:

- ako se u nizu nalaze eksplozije, sve eksplodiraju i nastaje niz spajanjem ostataka;
- spajanjem je moguće da su nastale nove eksplozije;
- lančana reakcija se ponavlja.

Mario se previše zaigrao i nije primijetio da je prišao vatri. Postalo je prekasno da bježi, te je odlučio da sazna koji će to niz da ostane nakon lančane reakcije. U slučaju da u nizu nije ostalo ništa, potrebno je ispisati “FRULA” (bez navodnika). U suprotnom, potrebno je ispisati izgled niza nakon završene lančane reakcije.

**Napomena:** U eksploziji se neće pojaviti dva ista znaka.

### Ulazni podaci

U prvom redu se nalazi Marioov **niz**.

U drugom red se nalazi riječ koja predstavlja **eksploziju**.

Mariov niz i eksplozija se sastoje od velikih i malih slova engleske abecede, te cifara 0, 1, ..., 9.

### Ograničenja

$1 \leq |\text{niz}| \leq 1\,000\,000$  - odnosi se na dužinu niza,

$1 \leq |\text{eksplozija}| \leq 62$  - odnosi se na dužinu riječi koja predstavlja eksploziju.



## Bodovanje

U testnim primjerima, za 50% slučajeva će vrijediti da je  $|niz| \leq 3000$ .

## Izlazni podaci

U prvi i jedini red ispisati konačan izgled Mariovog niza nakon lančane reakcije opisane u zadatku.

## Primjer

Ulaz	Očekivani izlaz	Objašnjenje
mariovC4nizCC44 C4	mariovniz	Prvo jednom eksplodira izraz "C4" pri čemu ostaje niz "mariovnizC4". Potom se još jednom dešava reakcija, pri čemu nam preostaje samo "mariovniz".
12ab112ab2ab 12ab	FRULA	Eksplodiraju podnizovi na mjestima 1 i 6, pri čemu nam ostaje niz "12ab", nakon čega ponovo dolazi do eksplozije pri čemu u nizu ne preostaje ništa.



## Zadatak 4 - Univerziteti

---

Auroria je država u kojoj postoji  $n$  gradova povezanih sa  $n - 1$  dvosmjernim putevima tako da je moguće doći iz bilo kojeg grada u bilo koji drugi grad. U Auroria-i postoji  $2k$  univerziteta koji se nalaze u različitim gradovima.

Nedavno je predsjednik potpisao ugovor o povezivanju univerziteta brzom mrežom. Ministarstvo prosvjete je ugovor shvatilo na svoj način i odlučilo da je dovoljno da se svaki univerzitet poveže kablom sa nekim drugim. Formalno, ugovor će biti ispoštovan!

Da bi osigurali maksimalan iznos u budžetu, Ministarstvo je odlučilo da univerzitete podijeli u parove tako da ukupna dužina potrebnog kabla bude maksimalna. Drugim riječima, ukupna udaljenost između univerziteta u  $k$  parova treba biti što veća.

Pomozite Ministarstvu da pronađe maksimalnu ukupnu udaljenost. Naravno, svaki univerzitet treba da bude prisutan samo u jednom paru. Uzmite u obzir da svi putevi imaju istu dužinu koja je jednaka 1.

### Ulazni podaci

Prvi red se sastoji od 2 cijela broja  $n$  i  $k$  - broj gradova u Auroria-i i broj univerzitetskih parova.

Uzeti u obzir da su gradovi numerisani brojevima od 1 do  $n$ .

Drugi red se sastoji od  $2k$  različitih cijelih brojeva  $u_1, u_2, \dots, u_{2k}$  ( $1 \leq u_i \leq n$ )- indeksi gradova u kojima su locirani univerziteti.

Narednih  $n - 1$  linija sadrži opise puteva. Svaka od linija sadrži par cijelih brojeva  $x_j$  i

$y_j$  ( $1 \leq x_j, y_j \leq n$ ), što znači  $j$ -ti put spaja gradove  $x_j$  i  $y_j$ . Kretanje od grada do grada se vrši samo pomoću ovih puteva.

### Ograničenja

$$2 \leq n \leq 200\,000,$$

$$1 \leq k \leq n/2.$$



Podzadatak 1 (25 bodova)

$$n \leq 100, 1 \leq k \leq n/2$$

Podzadatak 2 (5 bodova)

$$2 \leq n \leq 200\,000, k = 1$$

Podzadatak 3 (5 bodova)

$$2 \leq n \leq 200\,000, k = 2$$

Podzadatak 4 (25 bodova)

$$2 \leq n \leq 200\,000, k \leq 2000$$

Podzadatak 5 (40 bodova)

Bez dodatnih ograničenja.

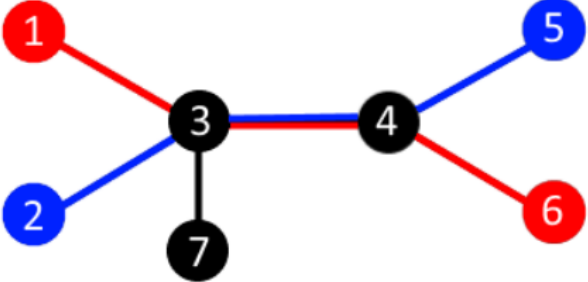
### Izlazni podaci

Ispišite maksimalnu moguću sumu udaljenosti u podjeli univerziteta na  $k$  parova.

### Primjer

Ulaz	Očekivani izlaz	Objašnjenje
7 2 1 5 6 2 1 3 3 2 4 5 3 7	6	Slika ispod prikazuje jednu od mogućih podjela u parove u prvom testu. Ako povežete univerzitete broj 1 i 6 (označeno crvenom bojom) i univerzitete broj 2 i 5 (označeno plavom) pomoću kabla, ukupna udaljenost će biti jednaka 6 što će biti maksimalni zbir u ovom primjeru.



4 3 4 6		
9 3 3 2 1 6 5 9 8 9 3 2 2 7 3 4 7 6 4 5 2 1 2 8	9	



## Zadatak 5 - Skakač

---

Tarik i Boris igraju najnoviju igru poznatiju kao Skakač. Tarik postavi figuru skakača na proizvoljno polje na  $N \times N$  šahovskoj ploči, te zatim, nakon što Boris zatvori oči, odigra točno  $T$  poteza, svake sekunde po jedan. Nakon što Tarik odigra posljednji potez, Boris mora, kako bi pobijedio, pogoditi gdje se figura nalazi. Šahovska ploča u ovoj igri specifična je po tome što za svako polje postoji određen broj  $K$  te je ono slobodno u sekundama  $0, K, 2K, 3K, \dots$  a ostatak vremena blokirano pa se skakač na njemu ne može nalaziti u tim sekundama. Igra počinje u nultoj sekundi, a Tarik svake sekunde mora pomaknuti skakača na jedan od 8 načina (kao u šahu, pomak skakača je u obliku slova "L", dva polja u jednom smjeru i jedno u drugom) pod uvjetom da to polje nije blokirano sljedeće sekunde. Pomozite Borisu i napišite program koji će ispisati sva polja na kojima se skakač može nalaziti nakon što Tarik odigra  $T$  poteza.

### Ulazni podaci

U prvom redu ulaza nalaze se dva prirodna broja  $N$ , dimenzija šahovske ploče i  $T$ , broj poteza koje će Tarik napraviti. U drugom redu nalaze se dva prirodna broja  $X$  i  $Y$ , oznaka reda i kolone početnog polja koje je Tarik odabrao. U sljedećih  $N$  redova nalazi se po  $N$  prirodnih brojeva koji predstavljaju vrijednosti  $K$  za odgovarajuća polja šahovske ploče.

### Ograničenja

$$3 \leq N \leq 30$$

$$1 \leq T \leq 1\,000\,000$$

$$1 \leq X, Y \leq N$$

Vrijednosti  $K$  za polja šahovske ploče su prirodni brojevi manji od  $10^9$  (miliardu)

### Bodovanje

U testnim primjerima, za 40% slučajeva broj poteza  $T$  će biti manji od 50 000.



## Izlazni podaci

U prvi red izlaza potrebno je ispisati M, broj polja na kojima se skakač može nalaziti na kraju igre. U sljedećih M redova potrebno je ispisati ta polja, uzlazno sortirana prema oznaci reda, a polja istog reda prema oznaci kolone.

### Primjer

Ulaz	Očekivani izlaz	Objašnjenje																											
3 2 1 1 1 3 2 2 3 2 3 1 1	2 1 1 1 3	<p>Za svaku sekundu prikazana je šahovska ploča, velikim slovom S označena su slobodna polja na kojima se skakač mogao nalaziti te sekunde, znakom # blokirana polja a tačkom slobodna.</p> <div><table><tr><td>S</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td></tr></table><table><tr><td>.</td><td>#</td><td>#</td></tr><tr><td>#</td><td>#</td><td>#</td></tr><tr><td>#</td><td>S</td><td>.</td></tr></table><table><tr><td>S</td><td>#</td><td>S</td></tr><tr><td>.</td><td>#</td><td>.</td></tr><tr><td>#</td><td>.</td><td>.</td></tr></table></div>	S	.	.	.	.	.	.	.	.	.	#	#	#	#	#	#	S	.	S	#	S	.	#	.	#	.	.
S	.	.																											
.	.	.																											
.	.	.																											
.	#	#																											
#	#	#																											
#	S	.																											
S	#	S																											
.	#	.																											
#	.	.																											
3 3 2 2 3 6 4 2 2 5 1 3 7	0	Nema polja na kojem se skakač može nalaziti na kraju igre.																											





5 6	5	
2 3	1 4	
4 5 3 2 3	2 1	
1 3 4 3 1	2 5	
3 4 1 3 2	4 5	
4 4 2 1 3	5 2	
4 6 4 9 2		



# Rješenja zadataka

Sva rješenja, tekstovi zadataka kao i testni slučajevi su (ili će biti) dostupni na linku: [https://github.com/hhadzem/ks\\_inf\\_takm](https://github.com/hhadzem/ks_inf_takm).

## Zadatak 1 - Portal

---

```
#include <iostream>

using namespace std;

int main() {
    int n, niz[100];
    cin >> n;

    for(int i = 0; i < n; i++) {
        cin >> niz[i];
        if(niz[i] % 2 + i % 2 != 1) {
            cout << "Portal ostaje zatvoren";
            return 0;
        }
    }

    cout << "Portal se otvara";
    return 0;
}
```



## Zadatak 2 - Tastatura

---

```
#include <iostream>
#include <utility>

using namespace std;

int main() {
    int n;
    char iz[1000], u[1000];
    string rijec;
    char mjesto[1000];

    for(int i = 'a'; i <= 'z'; i++)
        mjesto[i] = i;

    cin >> n;
    for(int i = 0; i < n; i++) {
        cin >> iz[i] >> u[i];
        swap(mjesto[iz[i]], mjesto[u[i]]);
    }
    cin >> rijec;

    for(int i = 0; i < rijec.size(); i++)
        cout << mjesto[rijec[i]];

    return 0;
}
```



## Zadatak 3 - Eksplozija

---

```
#include <stdio>
#include <cstring>
#include <stack>

#define MAXN 1000000
#define MAXM 62

#define from first
#define at second

using namespace std;

typedef pair<int, int> track;

char line[MAXN + 1];
char bomb[MAXM + 1];
int begin_cut[MAXN], end_cut[MAXN];
int n, m, left;

void bomb1(void){
    for (int i=0; i<n; i++){
        if (line[i] == bomb[0]) { begin_cut[i]++; end_cut[i]++; }
    }
}

void bomb2(void){
    stack<track> S;
    for (int i=0; i<n; i++){
        if (line[i] == bomb[0]) { S.push(track(i, 1)); continue; }

        while(!S.empty()){
            track tmp = S.top(); S.pop();
            if (line[i] != bomb[tmp.at]) {
                while(!S.empty()) S.pop();
                break;
            }
        }

        tmp.at++;
        if (tmp.at == m) { begin_cut[tmp.from]++; end_cut[i]++; }
    }
}
```



```

        else S.push(track(tmp.from, tmp.at));
        break;
    }
}

void remove(void){
    int erase = 0; left = 0;
    for (int i=0; i<n; i++){
        erase += begin_cut[i];

        if (erase) line[i] = '*';
        else left++;

        erase -= end_cut[i];
    }
}

void write(void){
    if (left){
        for (int i=0; i<n; i++) if (line[i] != '*') printf ("%c", line[i]);
    } else printf ("FRULA");
    printf ("\n");
}

int main(void){
    scanf ("%s%s", line, bomb);

    n = strlen(line), m = strlen(bomb);

    if (m == 1) bomb1();
    else        bomb2();

    remove();
    write();

    return 0;
}

```



## Zadatak 4 - Univerzitet

---

```
#include <iostream>
#include <vector>
#define pb push_back
using namespace std;
void print(vector<int> vi)
{
    for (auto t : vi) cout << t << ' ';
    cout << endl;
}

const int N = 2e5 + 10;
vector<int> g[N], f(N);
int ans;
int n, k;

void dfs(int u, int fa) {
    for (int v: g[u]) {
        if (v == fa) continue;
        dfs(v, u);
        f[u] += f[v];
        ans += min(f[v], k-f[v]);
    }
}

void solve()
{
    cin >> n >> k;
    k <= 1;
    for (int i = 0; i < k; i++) {
        int x;
        cin >> x;
        f[x] = 1;
    }
}
```



```

    for (int i = 1; i < n; i++) {
        int u, v;
        cin >> u >> v;
        g[u].pb(v);
        g[v].pb(u);
    }

    dfs(1, -1);
    cout << ans << endl;
}

int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL); cout.tie(NULL);

    int T = 1;
    while (T--) {
        solve();
    }
    return 0;
}

```



## Zadatak 5 - Skakač

---

```
#include <cstdio>
#include <cstring>
#include <algorithm>
#include <vector>

using namespace std;

int dx[] = { -2, -2, -1, -1, 1, 1, 2, 2 };
int dy[] = { -1, 1, -2, 2, -2, 2, -1, 1 };

const int MAXN = 31;
const int MAXT = 1000001;
const int MAXP = 170; //prosti brojevi manji od 1000
const int MAXPOT = 20; //najveca potencija nekog prostog broja u rastavu broja t

int pr[MAXT], f[MAXT], ff[MAXT], fp[MAXT];
int k[MAXN][MAXN], b[MAXN], bb[MAXN], m[MAXN];
int n, T;

int ind[1000], prosti[MAXP], cnt;
int faktor[MAXP][MAXPOT][MAXN]; // matrica F(p na q)
int fnula[MAXP][MAXP][MAXN]; // matrica G(x,y)

vector< int > v[MAXT];

inline void postavi( int &x, int y, int v ) { // postavlja y-ti bit u x na
vrijednost v
    if( v == 1 ) x |= (1<<y); else
        x &= ~(1<<y);
}

inline int daj( int x, int y ) { return (x>>y)&1; } // vraca vrijednost y-tog bita
u x

void sito( int MAX ) {
// izracunava sve proste brojeve manje od 1000
    for( int i = 2; i*i < MAXT; ++i )
        if( !pr[i] ) {
```





```

        for( int j = i+i; j < MAXT; j += i )
            pr[j] = 1, f[j] = i;
        ind[i] = cnt;
        prosti[cnt++] = i;
    }

    // izracunava po jedan prosti faktor i njegov eksponent za svaki broj manji od MAXT
    // kako bi mogli brojeve rastavljati na proste faktore
    for( int i = 2; i < MAXT; ++i ) {
        if( f[i] == 0 ) f[i] = i;
        int x = i;
        ff[i] = 1, fp[i] = 0;
        while( x%f[i] == 0 ) x /= f[i], ff[i] *= f[i], fp[i]++;
    }
}

void solveG() { // izracunava funkciju G(i,j)
    for( int i = 0; i < cnt; ++i ) {
        for( int j = 0; j < n; ++j )
            fnula[i][i][j] = faktor[i][0][j];

        for( int j = i+1; j < cnt; ++j )
            for( int k = 0; k < n; ++k )
                fnula[i][j][k] = (fnula[i][j-1][k] & faktor[j][0][k] );
    }
}

int main( void ) {
    scanf( "%d %d", &n, &T );

    int px, py;
    scanf( "%d %d", &px, &py ); --px, --py;

    sito( MAXT );

    for( int i = 0; i < n; ++i )
        for( int j = 0; j < n; ++j ) {
            scanf( "%d", k[i]+j );
            if( k[i][j] >= 1000 ) { // manje od 1000 slobodnih vremena, sve ih stavi
                // ručno
                for( int l = k[i][j]; l <= T; l += k[i][j] )
                    v[l].push_back( i*32 + j );
            } else { // period manji do 1000

```



```

    int x = k[i][j];
    for( int p = 0; p < cnt; ++p ) {
        int c = 0;
        while( x%prosti[p] == 0 ) x /= prosti[p], c++;
        while( c < MAXPOT ) postavi( faktor[p][c][i], j, 1 ), c++;
    }
}
}

solveG();

postavi( b[px], py, 1 );
int ms1 = (1<<(n-1))-1, ms2 = (1<<(n-2))-1;

for( int t = 0; t < T; ++t ) {
    // b - trenutna matrica
    // bb - sljedeca matrica
    // m - matrica blokiranosti sekunde t+1

    for( int i = 0; i < n; ++i ) {
        if( i-1 >= 0 ) bb[i-1] |= (b[i]>>2)|((b[i]&ms2)<<2);
        if( i-2 >= 0 ) bb[i-2] |= (b[i]>>1)|((b[i]&ms1)<<1);
        if( i+1 < n ) bb[i+1] |= (b[i]>>2)|((b[i]&ms2)<<2);
        if( i+2 < n ) bb[i+2] |= (b[i]>>1)|((b[i]&ms1)<<1);
    }

    for( int i = 0; i < n; ++i ) {
        b[i] = bb[i], bb[i] = 0;
        m[i] = (1<<n)-1;
    }

    int x = t+1, last = cnt-1;
    while( x > 1 && f[x] < 1000 ) {
        int now = ind[ f[x] ];

        if( last >= now+1 ) {
            for( int i = 0; i < n; ++i )
                m[i] &= fnula[now+1][last][i];
        }
        for( int i = 0; i < n; ++i )
            m[i] &= faktor[now][ fp[x] ][i];
        x /= ff[x], last = now-1;
    }
}

```



```

if( last >= 0 )
    for( int i = 0; i < n; ++i )
        m[i] &= fnula[0][last][i];

for( int i = 0; i < v[t+1].size(); ++i )
    postavi( m[ v[t+1][i]/32 ], v[t+1][i]%32, 1 );

for( int i = 0; i < n; ++i )
    b[i] &= m[i];
}

int ans = 0;
for( int i = 0; i < n; ++i )
    for( int j = 0; j < n; ++j )
        if( daj( b[i], j ) ) ans++;

printf( "%d\n", ans );
for( int i = 0; i < n; ++i )
    for( int j = 0; j < n; ++j )
        if( daj( b[i], j ) ) printf( "%d %d\n", i+1, j+1 );
return 0;
}

```

