Infokup

Županijsko natjecanje / Algoritmi /
Srednja škola (I. Podskupina)

Agencija za odgoj i obrazovanje
Education and Teacher Training Agency

MINISTARSTVO ZNANOSTI, OBRAZOVANJA
I ŠPORTA REPUBLIKE HRVATSKE

udruga mladih programera
dump

Sponzori Infokupa

Glavni sponzor

Mali sponzori

Medijski pokrovitelji

Sadržaj

Upute za natjecatelje

Kod svakog pojedinog zadatka obratite pozornost na poglavlja **ulazni podaci** i **izlazni podaci**. Tu su definirana pravila vezana za format ulaznih i izlaznih podataka koji mora biti strogo poštovan kako bi vaša rješenja bila točno evaluirana. Za ilustraciju i bolje razumijevanje pogledajte poglavlje p*rimjer pravilno napisanog programa*.

Ulaz i izlaz treba se odvijati preko standardnog ulaza i standardnog izlaza (to znači ***cin***, ***cout***, ***printf*** i/ili ***scanf***). Vaš program sa standardnog ulaza (***cin*** ili ***scanf***) mora očekivati samo ulazne podatke, a na standardni izlaz (***cout*** ili ***printf***) mora ispisivati samo izlazne podatke. Ako vaš program bude čekao na unos nečeg drugog osim ulaznih podataka ili ispisivao nešto drugo osim izlaznih podataka(npr. "Unesite brojeve...", "Rješenje je..." i slično), nećete dobiti bodove za taj zadatak, jer evaluator to ne očekuje. U pisanju programa dozvoljeno je koristiti samo standardne biblioteke, pa je tako primjerice zabranjeno koristiti naredbe ***clrscr();*** ili ***getch();*** jer su one dio ***conio.h*** *biblioteke koja nije standardna u jeziku* ***C/C++.***

Važno je napomenuti da ne trebate kreirati izvršnu datoteku (.exe), jer će ju sustav sam kreirati iz izvornog koda na sljedeći način:

- **pascal**: fpc -O1 -XS –o*PRG*.exe *PRG*.pas
- **C**: gcc -o *PRG*.exe *PRG*.c -std=c99 -O2 -s -static -lm
- **C++**: g++ -o *PRG*.exe *PRG*.cpp -O2 -s -static –lm

  (gdje je PRG ime programa)

Računalo na kojem se izvode programi i mjerena su vremenska ograničenja je Linux računalo s 2GHz radnog takta procesora.

Vaš program treba biti pisan u programskom jeziku C, C++ ili Pascal i mora regularno završiti svoje izvođenje. Program se treba izvršiti do kraja tj. do ***return 0***; na kraju funkcije 'main' koja treba biti deklarirana kao ***int main()***, ili naredbom ***exit(0);***. Pogledajte priložene primjere. U programskom jeziku pascal program se treba izvršiti do kraja tj. Do ***'end.'*** ili naredbom ***halt(0);***. Vaši programi **ne smiju pristupati** nikakvim datotekama **niti ih kreirati**, kršenje ovog pravila rezultirati će gubitkom bodova za taj zadatak. Bilo kakav pokušaj **pristupanja sistemskim datotekama računala na kojem se nalazi evaluator ili pokušaj upravljanja tim računalom** rezultirat će **diskvalifikacijom** tog natjecatelja. **Za dodjelu bodova važan je samo točan ispis rezultata.** Prilikom evaluacije nitko neće gledati vaš izvorni kôd već će on samo biti korišten za izradu izvršne datoteke, a bodove za pojedini test podatak će dobiti samo oni programi koji budu generirali **točan rezultat unutar navedenog vremena i memorijskog ograničenja**. Obratite pažnju da svi zadaci ne nose jednak broj bodova. Lakši i brže rješivi zadaci nose manje bodova, a teži zadaci za čije je rješavanje potrebno više vremena, znanja i koncentracije nose više bodova.

Prilikom rješavanja zadataka preporučuje se korištenje olovke i papira za skiciranje i razradu algoritma.

Primjer pravilno napisanog programa

Zadatak:

Napišite program koji će zbrojiti i oduzeti dva cijela broja.

Ulaz: U prvom retku se nalaze dva cijela broja A i B, međusobno odvojena jednim razmakom.

Izlaz: U prvi redak ispišite zbroj, a u drugi redak razliku brojeva A i B.

Rješenje u programskom jeziku C

```
#include <stdio.h>

int main()
{
    int a, b;
    scanf("%d%d", &a, &b);
    printf("%d\n", a+b);
    printf("%d\n", a-b);
    return 0;
}
```

Rješenje u programskom jeziku C++

```
#include <iostream>
using namespace std;

int main()
{
    int a, b;
    cin >> a >> b;
    cout << a+b << endl;
    cout << a-b << endl;
}
```

Rješenje u programskom jeziku Pascal

```
program p(input,output);
var
    a,b : integer;
begin
    read(a,b);
    writeln(a+b);
    writeln(a-b);
end.
```

Zadaci

U tablici možete pogledati ograničenja za zadatke:

Zadatak	Zadaća	Epruveta	Robot	Mreža
Naziv izvornog kôda	zadaca.cpp zadaca .c zadaca .pas	epruveta.cpp epruveta.c epruveta .pas	robot.cpp robot.c robot .pas	mreza.cpp mreza.c mreza .pas
Ulazni podaci	Standardni ulaz	Standardni ulaz	Standardni ulaz	Standardni ulaz
Izlazni podaci	Standardni izlaz	Standardni izlaz	Standardni izlaz	Standardni izlaz
Vremensko ograničenje	1 sekunda	1 sekunda	1 sekunda	1 sekunda
Memorijsko ograničenje	32 MB	32 MB	32 MB	32 MB
Broj bodova	**20**	**40**	**60**	**80**
Ukupno bodova		**200**		

Zadatak: Zadaća

Vrem. ograničenje: 1 sekunda / 20 bodova / Mem. ograničenje: 32 MB

Pero je danas u školi učio cijele brojeve i operacije među njima, te je za domaću zadaću dobio jedan zadatak. Zadatak sadrži tri cijela broja i računske operacije među njima. Međutim, Pero nije dobro prepisao zadatak s ploče. Uspio je prepisati samo brojeve, ali mu nedostaju znakovi računskih operacija.

Pero ne zna što će učiniti, pa se odlučio malo poigrati sa zadatkom. S obzirom da su do sada u školi naučili samo dvije operacije – zbrajanje i množenje, Peru zanima koliki je najveći mogući rezultat koji može dobiti koristeći ta **tri zadana cijela broja** i operacije **zbrajanja i množenja**.

Pero želi da zadatak ostane što sličniji originalnome, pa ne želi mijenjati redoslijed brojeva, a kako još nisu učili korištenje zagrada, Pero neće koristiti ni zagrade.

Vaš zadatak je napisati program koji za zadana tri cijela broja ispisuje najveći mogući rezultat koji se može dobiti od ta tri broja uz pomoć operacija zbrajanja i množenja, bez mijenjanja redoslijeda brojeva i upotrebe zagrada.

Ulaz:

U prvom i jedinom retku se nalaze tri cijela broja **a**, **b**, i **c** (-1 000 ≤ a,b,c ≤ 1 000)

Izlaz:

U prvom i jedinom retku ispišite najveći mogući rezultat koji Pero može dobiti koristeći brojeve a, b, i c i operacije zbrajanja i množenja.

Test podaci:

	Test 1	Test 2	Test 3
Ulaz	19 -27 49	36 93 -98	-127 -24 86
Izlaz	41	3250	262128
Napomene		(1)	

(1) ***Objašenjenje drugog test primjera:*** *Najveći mogući rezultat se dobije ako Pero postavi računske operacije na sljedeći način: 36 * 93 + (-98)*

Zadatak: Epruvete

Vrem. ograničenje: 1 sekunda / 40 bodova / Mem. ograničenje: 32 MB

Marko je zaljubljenik u kemiju i upravo je kupio novi set epruveta. Marko je oduševljen novim epruvetama, a još ga više oduševljavaju dvije stvari - to što su epruvete **različitih polumjera** i to što su epruvete **beskonačno visoke**.

Kao i svaki pravi kemičar, prije upotrebe kemikalija, Marko je odlučio testirati svoje epruvete pomoću obične vode.

Marko testira epruvete na način da poreda svih n epruveta jednu do druge i označi ih brojevima od 1 do n. Nakon toga svaku epruvetu napuni vodom do određene visine, te nasumično odabere dvije epruvete i prelije sav sadržaj iz jedne u drugu. Proces slučajnog odabira dviju epruveta i prelijevanja vode ponovi M puta.

Nakon svih prelijevanja, Marka zanima kolika je visina vode u svakoj pojedinoj epruveti.

Međutim, Marko, kao i svaki pravi kemičar, želi biti jako precizan u svojim mjerenjima, pa visinu vode želi predstaviti kao **cijeli broj**, **potpuno skraćeni razlomak** ili **potpuno skraćeni parcijalni razlomak** (ukoliko visina nije cijeli broj).

Pomozite Marku i napišite program koji za zadanih N epruveta i M prelijevanja izračuna visine vode u svim epruvetama na kraju testa.

Ulaz:

U prvom retku se nalazi prirodni broj **N** ($1 \leq N \leq 10\,000$) koji označava broj epruveta koje je Marko kupio.

U svakom od sljedećih **N redaka** nalaze se po dva prirodna broja $\mathbf{R_i}$ i $\mathbf{H_i}$ ($1 \leq R \leq 1\,000$, $1 \leq H \leq 1\,000$) koji označavaju polumjer i-te epruvete i visinu vode u njoj.

U sljedećem retku se nalazi jedan prirodan broj **M** ($1 \leq M \leq 10\,000$) koji označava broj prelijevanja koje Marko želi izvesti.

U svakom od sljedećih **M redaka** nalaze se po dva prirodna broja $\mathbf{X_j}$ i $\mathbf{Y_j}$ ($1 \leq X,Y \leq N$) koji označavaju redne brojeve epruveta između kojih se vrši j-to prelijevanje. Sva voda se prelije iz epruvete X u epruvetu Y.

Napomena: volumen valjka iznosi $V = r^2 * h * \pi$ (gdje je V volumen, r radijus baze, a h visina valjka)

Izlaz:

U prvih **N redaka** ispišite visinu vode u svakoj pojedinoj epruveti nakon svih izvršenih prelijevanja, počevši od prve, pa sve do n-te epruvete.

Visine vode ispisati na sljedeći način:

- Ukoliko je visina vode cijeli broj, ispišite samo taj broj,
- Ukoliko visina vode nije cijeli broj i manja je od 1 tada ispišite visinu vode u obliku potpuno skraćenog razlomka (2/3, 7/9, 23/44),
- Ukoliko visina vode nije cijeli broj i veća je od 1 tada ispišite visinu vode u obliku potpuno skraćenog parcijalnog razlomka (7 3/5, 2 11/16, 1 14/17). Cijeli broj i razlomak moraju biti odvojeni jednim razmakom.

Test podaci:

	Test 1	Test 2	Test 3
Ulaz	3 3 6 4 3 2 5 2 1 3 3 1	5 3 6 6 4 1 8 6 6 5 1 5 2 5 3 1 5 3 4 3 2 3	7 3 6 20 1 3 5 2 2 4 3 3 7 1 5 6 2 3 1 4 5 4 4 1 7 2 5 4
Izlaz	8 2/9 3 0	6 8/9 0 385 0 0	12 2/9 1/80 49 4/9 0 0 7 0

Napomene

Zadatak: Robot

Vrem. ograničenje: 1 sekunda / 60 bodova / Mem. ograničenje: 32 MB

Tajna špijunska agencija „K.A.U.C.H.“ u svom skrivenom laboratoriju razvija novog špijunskog robota. Za sada su razvili samo prototip tog robota, te su mu dali kodno ime Nikola.

Od svih planiranih naprednih funkcija, robot Nikola ima implementirano samo napredno izbjegavanje prepreka. Algoritam za izbjegavanje prepreka osmislila je Lucija – glavna inženjerka agencije „K.A.U.C.H.“.

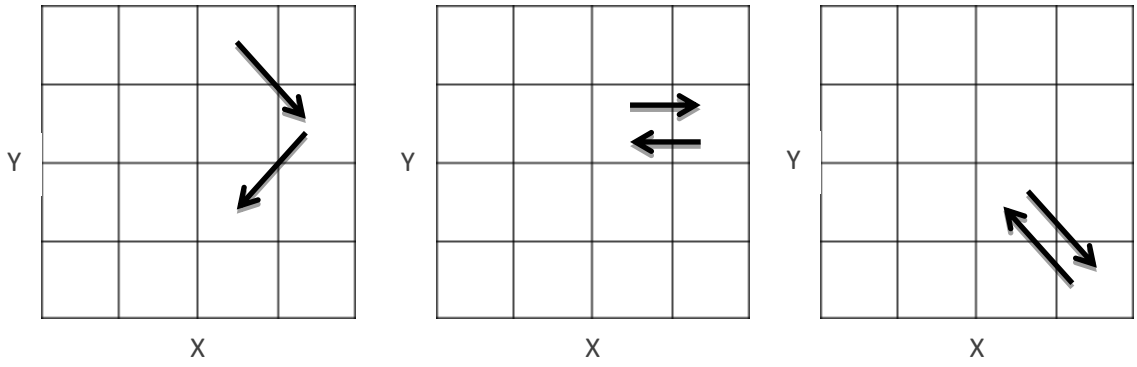
U agenciji trenutno traju pripreme za opširna i intenzivna testiranja najnovije verzije robota Nikola. Testirat će se kretanje robota, pa su inženjeri na dio poda u svojoj radionici ucrtali **koordinatni sustav** s označenim kvadratnim poljima od po jedan metar. Uz to su niskim zidovima ogradili jedan **pravokutni dio** u kojem će se roboti kretati. Zidovi su paralelni s koordinatnim osima, ishodište koordinatnog sustava nalazi se u donjem lijevom uglu pravokutnog dijela, a raspored osi je kao u kartezijevom koordinatnom sustavu.

Nikola se kreće u jednom od **osam smjerova** (paralelno s osima ili dijagonalno) brzinom od **jednog polja u sekundi**, a smjer kretanja mu zadaje Lucija na početku testa. Dakle, robot se iz trenutnog polja može pomaknuti u jedno od osam susjednih polja (ovisno o zadanom smjeru).

Algoritam izbjegavanja prepreka (u ovom slučaju zidova) radi na način da Nikola u svakom trenutku zna gdje se mora pomaknuti. Ukoliko bi neki pomak izazvao sudar sa zidom, Nikola prvo **promjeni smjer**, pa tek onda izvrši pomak.

Nikola mijenja smjer na sljedeći način:

- Ako će pod kutom udariti u zid, tada će promijeniti smjer za 90°
- Ako će okomito udariti u zid, promijenit će smjer za 180°
- Ako će u jednom od dijagonalnih smjerova udariti u kut, promijenit će smjer za 180°



S obzirom da je ova faza testiranja iznimno važna, Lucija želi biti sigurna da su testovi točni. Odredila je Nikolinu početnu točku i smjer kretanja. Lucija će unijeti niz nasumično odabranih vremena za koje želi znati gdje se točno nalazi Nikola. Vremena ne moraju biti posložena po redu.

Vaš zadatak je napisati program koji će odgovoriti na Lucijina pitanja.

Ulaz:

U prvom retku se nalaze dva prirodna broja **V** i **S** (2 ≤ V,S ≤ 500) koji predstavljaju visinu i širinu pravokutnog područja u kojem se Nikola kreće.

U drugom retku se nalaze dva broja i jedna riječ odvojeni razmacima. Brojevi **X** i **Y** predstavljaju koordinate Nikoline početne pozicije, unutar zadanog područja, a riječ se sastoji od dva velika slova engleske abecede i označava početni smjer kretanja robota.

Riječ se sastoji od dva velika slova engleske abecede, a sigurno će biti jedna od osam sljedećih:

„**UU**“	gore	„**DD**“	dolje
„**RU**“	gore desno	„**LD**“	dolje lijevo
„**RR**“	desno	„**LL**“	lijevo
„**RD**“	dolje desno	„**LU**“	gore lijevo

U trećem retku se nalazi jedan prirodni broj **N** (1 ≤ N ≤ 100 000) koji označava broj Lucijinih upita.

U sljedećem, četvrtom retku nalazi N prirodnih brojeva $\mathbf{T_i}$ (1 ≤ T ≤ 26 483 109) odvojenih razmakom, od kojih svaki predstavlja Lucijin upit za Nikolinu poziciju u vremenu T_i.

NAPOMENA: u 30% test podataka upiti će biti za vremena manja od 1000, a dimenzije pravokutnog područja će biti manje 50.

Izlaz:

U svaki od prvih N redaka ispišite po dva broja $\mathbf{X_i}$ i $\mathbf{Y_i}$ koji predstavljaju odgovor na i-to Lucijino pitanje, odnosno predstavljaju Nikolinu poziciju u trenutku T_i.

Test podaci:

	Test 1	**Test 2**	**Test 3**
Ulaz	4 8 4 2 RR 3 4 10 12	3 5 3 2 RU 3 3 5 7	8 8 7 4 RD 4 2 6 9 11
Izlaz	8 2 2 2 2 2	4 1 2 3 2 1	7 2 3 4 2 7 4 7
Napomene			

Zadatak: Mreža

Vrem. ograničenje: 1 sekunda / 80 bodova / Mem. ograničenje: 32 MB

Informatička tvrtka HookMeUp je na javnom natječaju dobila posao umrežavanja važnih gradskih lokacija. Cilj im je provući mrežne kabele koji će **spajati sve lokacije**. Dvije lokacije se smatraju spojenima ako među njima postoji direktna veza barem jednim kabelom ili postoji indirektna veza, odnosno niz drugih lokacija preko kojih su te dvije lokacije spojene.

S obzirom da tvrtka želi uštedjeti na iskopavanju kanala i postavljanju kabela, planiraju sagraditi takvu mrežu da ukupna cijena izgradnje cijele mreže bude **što manja** (cijena je proporcionalna dužini kabela), a da ipak sve lokacije budu spojene.

Grad je pristao na ovakav plan izgradnje, ali je od tvrtke HookMeUp zatražio da im dostavi informaciju o **najvećoj mogućoj mrežnoj latenciji** između bilo koje dvije lokacije **u tako izgrađenoj** mreži. Mrežna latencija je vrijeme potrebno da informacija stigne s jedne lokacije do druge. Latencija je proporcionalna ukupnoj dužini kabela u jednoj vezi (bilo direktnoj ili indirektnoj) između dviju spojenih lokacija, pa stoga tvrtku zanima kolika će biti **najveća udaljenost** između bilo koje **dvije lokacije**, tako da naknadno mogu izračunati stvarnu latenciju.

Ulaz:

U prvom redu se nalazi prirodan broj N ($1 \leq N \leq 500$), koji predstavlja broj lokacija koje je potrebno spojiti. U sljedećih N redova nalaze se po dva cijela broja X i Y ($-500 \leq X, Y \leq 500$), koji predstavljaju koordinate i-te lokacije.

Izlaz:

U prvom i jedinom retku potrebno je ispisati najveću udaljenost između bilo koje dvije lokacije u gore spomenutoj mreži. Broj treba zaokružiti na dvije decimalne znamenke (čak i ako je rješenje cijeli broj, potrebno je ispisati decimalnu točku i potom dvije nule).

Test podaci:

	Test 1	Test 2	Test 3
Ulaz	2 1 1 2 2	5 1 1 -2 6 4 6 1 2 3 6	9 -5 -3 -2 -1 -2 4 2 4 1 1 2 -3 3 3 3 1 7 2
Izlaz	1.41	10.47	16.63
Napomene			*(1)*

(1) ***Objašnjenje trećeg test primjera***: *Tvrtka će izgraditi mrežu kao na slici, jer je njena cijena minimalna u odnosu na druge moguće mreže. Tada je najveća udaljenost od točke 3 do točke 1. Kada zbrojimo sve udaljenosti dobijemo: (3->4 = 4); (4->7 = 1.41); (7->8 = 2); (8->5 = 2); (5->2 = 3.61); (2->1 = 3.61) i konačno, dužina cijelog puta iznosi 16.63.*