Županijsko natjecanje iz informatike

Srednja škola

Prva podskupina (1. i 2. razred)

9. veljače 2018.

ime zadatka	**BITCOIN**	**CONNECT**	**VRTULJAK**
vremensko ograničenje	1 sekunda	1 sekunda	2 sekunde
memorijsko ograničenje	512 MiB	512 MiB	512 MiB
broj bodova	40	50	60
	150		

Ministarstvo znanosti i obrazovanja

Agencija za odgoj i obrazovanje

HRVATSKA ZAJEDNICA TEHNIČKE KULTURE

HRVATSKI SAVEZ INFORMATIČARA

Sastavni dio kompleta zadataka su i ove upute te uvodna stranica na kojoj se nalaze važni podatci o zadacima. Molimo vas da i jedno i drugo pažljivo pročitate. Na ostalim stranicama nalaze se tri zadatka. Prilikom rješavanja zadataka preporučuje se korištenje olovke i papira za skiciranje i razradu algoritma.

Nakon završetka natjecanja, članovi županijskih povjerenstava poslat će izvorne kodove vaših rješenja na automatsku evaluaciju. Nakon završetka evaluacije za vašu županiju, rezultati evaluacije bit će dostupni putem sustava za evaluaciju na adresi https://srednje.hsin.hr.

Žalbe na rezultate evaluacije možete uputiti direktno Županijskom povjerenstvu (koje će vas obavijestiti o roku i načinu predaje žalbi) ili Državnom povjerenstvu na adresu dp-informatika@azoo.hr. Rok za predaju žalbi Državnom povjerenstvu istječe *u ponoć na dan natjecanja*. Rok žalbi Državno povjerenstvo može dodatno produžiti u slučaju većih problema sa zadatcima ili sustavom za evaluaciju.

Ulazni i izlazni podatci

Kod svakog pojedinog zadatka obratite pozornost na sekcije *Ulazni podatci* i *Izlazni podatci*. Tu su definirana pravila vezana uz format ulaznih i izlaznih podataka koji mora biti strogo poštovan kako bi vaša rješenja bila ispravno evaluirana. Vaš program sa standardnog ulaza mora očekivati samo zadane ulazne podatke, a na standardni izlaz ispisivati samo tražene izlazne podatke bez ikakvih dodatnih poruka. 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 jer evaluator to ne očekuje.

Vaši programi ne smiju pristupati datotekama ili ih kreirati te ne smiju pokretati nove procese ili koristiti više od jedne dretve (threada).

Prilikom rješavanja nekog zadatka i testiranja njegovog rješenja preporučuje se korištenje operatora redirekcije ulaza kako ne biste više puta nepotrebno unosili podatke preko tipkovnice. Na primjer, ulazne podatke za neki od oglednih primjera iz teksta zadatka možete spremiti u tekstualnu datoteku i testirati vaš program tako da ga pokrećete iz komandne linije na sljedeći način (pretpostavimo da se zadatak zove „Neboder“):

```
neboder < primjer.txt
```

Znak < je operator redirekcije ulaza i sve što se nalazi u datoteci `primjer.txt` bit će proslijeđeno vašem programu kao da je uneseno preko tipkovnice.

U slučaju Pythona potrebno je eksplicitno pozvati odgovarajući prevoditelj. Na primjer:

```
C:\Python27\python neboder.py < primjer.txt
```

Bodovanje

Da bi program koji rješava neki zadatak dobio maksimalan broj bodova, primijenjeni algoritam mora biti najprije točan, ali i efikasan tj. brz. Test podatci unaprijed su osmišljeni tako da će programi koji koriste manje efikasne, ali valjane algoritme, također dobiti određeni broj bodova (npr. od ukupno 60 bodova, vrlo spor algoritam dobit će npr. 20 bodova, dok će dobar algoritam, ali ne i najbolji, dobiti npr. 40 bodova). Jako brz program koji ne daje točne rezultate, naravno, ne donosi bodove. Valjanost algoritma je na prvom mjestu, a brzina izvršavanja na drugom.

Različite zadatke možete rješavati u različitim jezicima, imajući na umu da će rješenja u Pythonu, zbog prirode jezika, biti sporija od ekvivalentnih rješenja u drugim dozvoljenim jezicima i da bi zato mogla na nekim test podatcima premašiti vremensko ograničenje.

Neki od službenih test podataka mogu biti grupirani na način da rješenje dobiva bodove za pojedinu grupu (*cluster*) samo ako uspješno riješi sve test podatke iz grupe.

Evaluacija

Bodove za pojedini test podatak dobit će samo oni programi koji budu generirali točan rezultat unutar navedenog vremenskog i memorijskog ograničenja, te regularno završe svoje izvođenje. Točnije, program se treba izvršiti do kraja. U programskim jezicima C/C++ to znači do `return 0;` na kraju funkcije `main` koja treba biti deklarirana kao `int main()`, ili do naredbe `exit(0);`.

Ne trebate predate izvršnu (exe) datoteku već samo predajete datoteku s izvornim kodom. Imena datoteka moraju odgovarati imenima zadataka, a ekstenzija datoteke mora odgovarati programskom jeziku i standardu u skladu s donjom tablicom (`.c, .cpp, .cxx` ili `.py`). Npr. ako se zadatak zove „Neboder“ i ako koristite C++11, predat ćete datoteku `neboder.cxx`. Na temelju odgovarajuće ekstenzije, za jezike C/C++ sustav za evaluaciju iz vašeg će izvornog kôda kreirati izvršnu datoteku na sljedeći način:

```
C          gcc -DEVAL -O2 -o neboder neboder.c -lm
C++        g++ -DEVAL -O2 -o neboder neboder.cpp
C++11      g++ -DEVAL -std=c++11 -O2 -o neboder neboder.cxx
```

Za rješenja u Pythonu, kako bi evaluator prepoznao koristite li verziju 2 ili verziju 3, prva linija u kodu mora točno odgovarati jednom od sljedećeg:

- `#!/usr/bin/python2`
- `#!/usr/bin/python3`

Za evaluaciju će se koristiti verzije prevoditelja 2.7 odnosno 3.5, a prije izvođenja vaš kod bit će preveden u bytecode naredbom `python -m py_compile neboder.py`.

Računalo na kojem se vrši evaluacija je Linux računalo s Ubuntu 16.04 LTS 64-bitnim operativnim sustavom i sljedećim verzijama prevoditelja: gcc 4.9, g++ 4.9. Preporučujemo da svoja rješenja obavezno isprobate sa `gcc` ili `g++` prevoditeljima s gore navedenim opcijama, osobito ako koristite neki drugi alat (npr. Microsoft Visual Studio) tijekom natjecanja. Da bi vaše rješenje bilo uspješno prevedeno, morate koristiti samo standardne biblioteke, tj. ne smijete koristiti naredbe i funkcije specifične za Windowse.

Primjeri pravilno napisanih programa

Zadatak: Napišite program koji će zbrojiti i oduzeti dva cijela broja.

Ulaz: U prvom retku nalaze se 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.

C

```
#include <stdio.h>
int main(void) {
  int a, b;
  scanf("%d%d", &a, &b);
  printf("%d\n", a + b);
  printf("%d\n", a - b);
  return 0;
}
```

C++

```
#include <iostream>
using namespace std;
int main(void) {
  int a, b;
  cin >> a >> b;
  cout << a + b << endl;
  cout << a - b << endl;
  return 0;
}
```

Python 2

```
#!/usr/bin/python2
a, b = map(int, raw_input().split())
print a + b
print a - b
```

Python 3

```
#!/usr/bin/python3
a, b = map(int, input().split())
print(a + b)
print(a - b)
```

Česte pogreške

Donosimo listu **čestih i nepotrebnih pogrešaka** na natjecanjima ovog tipa. Sve od sljedećeg može rezultirati time da će evaluator dodijeliti nula bodova vašem rješenju.

• Manjak inicijalizacije lokalnih varijabli: U jezicima C i C++ lokalne varijable (što uključuje i varijable deklarirane unutar funkcije `main`) ne inicijaliziraju se automatski te njihova početna vrijednost nije definirana. Obavezno inicijalizirajte vrijednost svih lokalnih varijabli, inače je moguće da vaš program dobro radi lokalno, a dobije nula bodova na sustavu za evaluaciju.

• Rješenje alocira premala polja: Ako ne alocirate dovoljno velika polja, moguće je da vaš program dobro radi za primjere iz teksta zadatka, a dobije nula bodova na evaluatoru. Na primjer, ako s ulaza trebate pročitati niz od najviše 1000 znakova, u jeziku C potrebno je alocirati polje od najmanje 1001 elementa (`char s[1001]`).

• Korištenje `conio.h` u jezicima C i C++. Korištenje sintakse, tipova i funkcija specifičnih za Microsoftove C i C++ prevoditelje, npr. tip podataka `int64`.

• Rješenja čekaju na pritisnutu tipku nakon ispisa rezultata, npr. zbog naredbe `system("pause")`.

• Rješenja ispisuju rezultate u pogrešnom formatu, na primjer, ispisuju dva broja svaki u svoj redak umjesto u istom retku.

• Rješenja ispisuju višak podataka na standardni izlaz, npr. debug informacije ili poruke poput "Upišite broj:", "Rješenje je:" i slično.

• Glavna funkcija u jezicima C i C++ definirana je kao void `main()` ili nema naredbe `return 0;`.

• Manjak potrebnih include direktiva u jezicima C i C++.

• Korištenje naredbi `itoa` (koje nema pod Linuxom) ili `atoi` (koja radi malo drugačije pod Linuxom). Umjesto njih preporučamo korištenje funkcija `sscanf` i `sprintf`.

• Rješenje alocira prevelika polja: Nepotrebno glomazna polja mogu prouzročiti da vaše rješenje prekorači memorijsko ograničenje i dobije nula bodova na evaluatoru. Primjerice, u jeziku C polje deklarirano sa `int x[1024][1024][100]` koristi 400MiB memorije.

Dodatno, natjecateljima koji koriste C i C++ savjetujemo da prilikom testiranja obavezno uključe optimizacijsku opciju koja se koristi na sustavu na evaluaciju (`-O2`) te opcije koji upozoravaju na moguće probleme u kodu (primjerice `-Wall -Wextra`). Niže je ilustracija takvog prevođenja jednog rješenja sa Županijskog natjecanja 2016. godine. Prvo upozorenje koje je prijavio prevoditelj je lažna uzbuna, dok drugo ukazuje na stvarnu grešku (neinicijalizirana varijabla) zbog koje je natjecatelj nepotrebno izgubio bodove.

```
$ g++ -O2 -o poli.exe poli.cpp -Wall -Wextra
poli.cpp: In function ‘int main()’:
poli.cpp:16:20: warning: comparison between signed and unsigned integer expressions [-Wsign-compare]
  for (int i = 0; i < in.length(); ++i)
                  ^
poli.cpp:36:29: warning: ‘curr’ may be used uninitialized in this function [-Wmaybe-uninitialized]
  num[(curr == 0 ? 1 : curr) - 1] = val;
```

Mladom Dominiku baka je dala 10 eura za sladoled. Umjesto da se udeblja i pokvari zube jedući sladoled, Dominik je kao pravi poduzetnik odlučio novac uložiti u kriptovalute. Njegov osobni bankar dostavio mu je predviđanja cijene bitcoina za idućih N dana, a Dominik je odlučio ulagati na sljedeći način:

- prvo će odabrati jedan dan kada će **svih 10 eura** uložiti u bitcoin,
- u jednom od dana nakon toga prodat će **sve** bitcoine za eure,
- konačno, u jednom od dana nakon toga opet će kupiti bitcoine (i pritom potrošiti **sve eure**).

Koliko najviše bitcoina Dominik može imati nakon izvršavanja svojega plana?

ULAZNI PODATCI

U prvom retku nalazi se prirodan broj N ($3 \leq N \leq 100\,000$), broj dana.

U svakom od sljedećih N redaka nalazi se jedan realan broj zaokružen na 3 decimale – cijena bitcoina toga dana (u eurima). Cijena neće biti manja od 0.01 EUR ni veća od 100 000 EUR.

IZLAZNI PODATCI

Ispišite jedan realan broj, najveći broj bitcoina koje Dominik može imati po izvršetku svog plana. Rezultat će se smatrati točnim ako odstupa od službenog za najviše 0.001.

BODOVANJE

U test podatcima ukupno vrijednima 30% bodova bit će $3 \leq N \leq 500$.

U test podatcima ukupno vrijednima 60% bodova bit će $3 \leq N \leq 5000$.

PRIMJERI TEST PODATAKA

ulaz

```
3
10.000
0.100
10.000
```

izlaz

```
0.01
```

ulaz

```
8
118.915
56.144
8.232
324.648
481.128
21.693
300.122
121.165
```

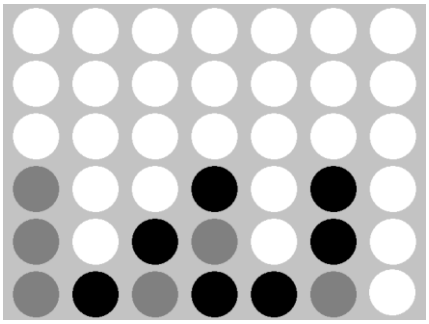
izlaz

```
26.942361
```

Pojašnjenje prvog primjera: Jedini način da Dominik provede svoj plan jest da kupuje prvi dan, prodaje drugi i opet kupuje treći dan. Prvi dan kupio je 1 bitcoin za 10 EUR, idući dan ga prodao za 0.1 EUR, i treći dan od tog novca kupio 0.01 bitcoin.

Pojašnjenje drugog primjera: Kao što vidimo, tržište valuta može biti nepredvidivo. Dominiku se najviše isplati kupiti treći dan, prodati peti i opet kupiti šesti dan.

"Connect Four" kombinatorna je igra dvaju igrača na okomitoj ploči sa šest redova i sedam stupaca, koja je na početku igre prazna, a tijekom igre u nju se ubacuju žetoni. Žeton se ubacuje u odabrani stupac ploče (ako u njemu ima slobodnih polja) i nakon ubacivanja žeton pada po tom stupcu do najnižeg slobodnog polja.



Na početku igre svaki igrač odabire boju svojih žetona. Tijekom igre, igrači naizmjence vuku poteze, ubacujući žetone svoje boje. Cilj je svakog igrača ostvariti niz od četiri žetona svoje boje koji su uzastopni u redu, stupcu ili po dijagonali. Igra završava čim se to dogodi i odgovarajući igrač proglašava se pobjednikom.

Vaš je zadatak za zadano stanje ploče, u koju su već ubačeni neki žetoni, odrediti postoji li potez kojim igrač na potezu pobjeđuje, te koji je to potez.

ULAZNI PODATCI

Ulaz se sastoji od šest redaka koji predstavljaju izgled ploče, redom od najvišeg do najnižeg reda. Svaki redak sadrži sedam znakova 0, 1 i 2 (bez razmaka). Znak 0 predstavlja prazno polje, znak 1 predstavlja žeton igrača na potezu, a znak 2 predstavlja žeton njegovog protivnika.

U skladu s gravitacijom, iznad praznog polja neće biti žetona. Na zadanoj ploči još nema pobjednika (ali ona možda i nije nastala naizmjeničnim potezima igrača).

IZLAZNI PODATCI

Ako igrač 1 ne može pobijediti u sljedećem potezu, ispišite broj -1. Inače, za svaki od mogućih pobjedničkih poteza, u zaseban redak izlaznih podataka ispišite redni broj stupca u koji treba ubaciti žeton. Potezi, tj. odgovarajući stupci trebaju biti ispisani u rastućem poretku.

PRIMJERI TEST PODATAKA

ulaz

```
0000000
0200201
0100101
0110201
0220102
0112101
```

izlaz

```
4
7
```

ulaz

```
0000000
0000000
1000000
1010021
2022021
2112212
```

izlaz

```
-1
```

Pojašnjenje prvog primjera: Ubacivanjem u 4. stupac igrač 1 ostvaruje niz od četiri svoja žetona dijagonalno, a ubacivanjem u 7. stupac ostvaruje niz od četiri svoja žetona u tom stupcu okomito.

U Wonkinoj tvornici čokolade glavna je atrakcija brzi vrtuljak uz koji se nalazi čokoladna prskalica. Dok se vrtuljak okreće, svake sekunde jedna osoba na vrtuljku prođe pokraj prskalice koja je tada poprska čokoladom.

U jednoj od današnjih vožnji na vrtuljku sudjelovalo je N djece, označenih brojevima od 1 do N redom kojim su sjedili na vrtuljku. Vožnja je trajala M sekundi. U sekundi 1 pokraj prskalice je prošlo dijete 1, u sljedećoj sekundi dijete 2, i tako dalje do djeteta N, nakon čega je u sekundi N + 1 ponovno dijete 1 prošlo pokraj prskalice, i tako dalje redom, u krug.

Prskalica je, nažalost, bila pokvarena i u nekim vremenskim intervalima nije prskala čokoladu pa su se djeca žalila upravitelju tvornice. Sakupljeno je K izjava sljedećeg oblika: „od sekunde A do sekunde B (uključivo) prskalica nije radila“. Napišite program koji, uz pretpostavku da su izjave istinite, za svako od N djece s vrtuljka određuje koliko je najviše puta to dijete moglo biti poprskano čokoladom tijekom vožnje na vrtuljku.

ULAZNI PODATCI

U prvom retku nalaze se prirodni brojevi N ($2 \leq N \leq 100\,000$) i M ($2 \leq M \leq 10^{16}$), broj djece na vrtuljku i duljina vožnje u sekundama.

U drugom retku nalazi se se prirodan broj K ($1 \leq K \leq 100\,000$), broj izjava.

U svakom od sljedećih K redaka nalaze se brojevi A i B ($1 \leq A \leq B \leq M$) koji znače da između sekundi A i B (uključujući i te sekunde) prskalica nije radila.

IZLAZNI PODATCI

Za svako dijete od 1 do N u zaseban redak ispišite traženi najveći mogući broj čokoladnih prskanja.

PRIMJERI TEST PODATAKA

ulaz

```
3 10
2
1 2
4 10
```

izlaz

```
0
0
1
```

ulaz

```
5 20
3
3 6
10 14
13 18
```

izlaz

```
1
2
1
2
1
```

ulaz

```
10 100
2
30 50
25 62
```

izlaz

```
6
6
7
7
6
6
6
6
6
6
```

Pojašnjenje prvog primjera: Prskalica nije radila u 1. i 2. sekundi, kao ni od 4. do 10. sekunde. Pretpostavimo li da je prskalica radila u 3. sekundi, dijete 3 moglo je tada biti poprskano. Ostalo dvoje djece nije moglo biti poprskano čokoladom.