Programmeringsolympiadens final 2010

TÄVLINGSREGLER

- Tävlingen äger rum den 5 eller 9 mars. Tävlingstiden är sex timmar effektiv tid.
- Tävlingen består av sju uppgifter som samtliga ska lösas genom datorprogram.
- Uppgifterna ska lösas i valfritt programmeringsspråk. Du får byta språk mellan olika uppgifter.
- Tävlingsbidragen lämnas som exekverbara filer i Windows-format (EXE). Dessutom ska källkoden bifogas. För Linux/Mac-användare som använder gcc kan vi göra undantag från regeln med EXE-filer och istället kompilera källkodsfilen själva. Inkludera en kommentar i källkoden med den fullständiga kommandoraden för att kompilera programmet. Detsamma gäller om du använder ett interpreterande språk, t.ex. PHP (ange även version). Om du använder Java, så fungerar class-filen som exekverbar.
- I varje källkodsfil ska finnas en kommentar innehållande namn och skola.
- Lösningarna poängsätts med max 5 poäng per uppgift. Fem tester, med varierande krav hos ditt program, kommer att göras vid rättningen (undantag kan finnas). Möjlighet till delpoäng finns om programmet klarar endast en del av dessa tester. Ingen närmare bedömning av programkoden görs.
- Ingen test av indata behöver göras. Alla testdata följer de specifikationer som givits i uppgiften. Om det trots detta, vid rättningen, uppstår exekveringsfel vid körning av programmet bedöms programmet som felaktigt för det testexemplet.
- Samtliga uppgifter leder fram till program vars exekveringstid bör understiga 5 sekunder på en modern dator. Skulle exekveringstiden för ditt program överskrida denna tid bedöms programmet med 0 poäng för detta testexempel.
- Du har tillgång till de indatafiler som används i uppgiftens exempel.
- Deltagandet är individuellt vilket bland annat innebär att inget utbyte av idéer eller filer får ske under tävlingstiden. Självklart får din dator inte vara kopplad till vare sig internt eller externt nät.
- Hjälpmedel förutom dator med installerade programspråk: Programspråkets manualer, vanlig formelsamling och räknedosa.
- I flera av uppgifterna ska indata läsas från en vanlig ASCII-fil (se nästa sida).
- Tävlingsbidragen ska läggas i roten på utdelat USB-minne eller i en av läraren angiven hårddiskkatalog. Filerna ska döpas till uppg1...uppg7 med passande filtillägg. Ingen hänsyn tas till andra filer. Var noga med att lämna in den korrekta versionen av ditt program.
- Tips: Det kan vara värt att göra egna indata för att testa ditt program. Även om programmet klarar testexemplet behöver det inte vara korrekt.

LATHUND FÖR INLÄSNING FRÅN FIL

Exempel på hur man i fem språk kan läsa in följande indata från filen `fil.txt`:

```
4 6
3.22 Text
```

Observera att fel kan förekomma.

C

```
#include <stdio.h>
...
int a1, a2;
char word[100];
double d;
FILE *fil=fopen("fil.txt", "rt");
fscanf(fil, "%d %d", &a1, &a2);
fscanf(fil, "%lf %s", &d, word);
```

C++

```
#include <iostream>
using namespace std;
...
int a1, a2;
char word[100];
double d;
ifstream fil("fil.txt");
fil >> a1 >> a2;
fil >> d >> word;
```

Java (J2SE)

```
import java.util.Scanner;
import java.io.File;
...
Scanner sc=null;
sc = new Scanner(new File("fil.txt"));
int a1=sc.nextInt(), a2=sc.nextInt();
double d=sc.nextDouble();
String word=sc.next();
```

Pascal

```
VAR
infile : Text;
a1, a2 : Integer;
d : Double;
word : string[100];
...
Assign(infile, 'fil.txt');
Reset(infile);
Readln(infile, a1, a2);
Readln(infile, d, word);
Close(infile);
```

Basic

```
Dim s,word As String
Dim a1, a2 As Integer
Dim d As Double
Dim sar() As String
...
Open "fil.txt" For Input As #1
Line Input #1, s
sar = Split(s," ")
a1 = val(sar(0))
a2 = val(sar(1))
Line Input #1, s
sar = Split(s," ")
d = val(sar(0))
word = sar(1)
```

Lycka till!

UPPGIFT 1 – TRUBADUREN

I en by på landet sjunger byborna varje kväll visor vid brasan. En viktig bybo är trubaduren. De kvällar trubaduren är närvarande sjunger han en ny visa som ingen annan bybo hört förut. Inga andra visor sjungs dessa kvällar, och alla bybor som då är närvarande lär sig trubadurens nya visa. De kvällar trubaduren inte är närvarande så sjunger byborna utan honom, och utbyter alla visor de kan med varandra.

Givet vilka bybor som är närvarade vid ett visst antal kvällar, skriv ut en lista på de bybor som efter dessa kvällar känner till samtliga visor som sjungits under den perioden.

Indata

Första raden i filen `trubadur.dat` innehåller heltalet N ($1 \leq N \leq 100$), antalet personer i byn, inklusive trubaduren. Byborna numreras från 1 till N, där bybo nummer 1 är trubaduren. Andra raden innehåller heltalet M ($1 \leq M \leq 50$), antalet kvällar det sjungs visor i byn. Därefter följer M rader som beskriver vilka bybor som närvarat de olika kvällarna. Varje sådan rad börjar med ett tal K som anger antalet bybor som var närvarande den kvällen. Därefter följer K tal som anger vilka dessa bybor var. Ingen bybo kommer nämnas mer än en gång per rad, och trubaduren kommer att närvara åtminstone en kväll.

Utdata

Skriv ut de bybor, inklusive trubaduren, som kan alla olika visor som sjungits efter de M kvällarna. Skriv ut byborna sorterade i nummerordning och separerade med blanksteg.

Exempel

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

Svar

```
1 2 4
```

Förklaring: Trubaduren är närvarande två kvällar, så det finns totalt två olika visor. Bybo 2 får lära sig den andra visan den tredje kvällan då trubaduren framför den och får lära sig den första visan via bybo 3 den andra kvällen. Bybo 4 får lära sig den ena visan kväll 2 och den andra kväll 4. Notera att bybo 5 aldrig får lära sig visan som sjöngs den första kvällen eftersom den inte sjöngs den tredje kvällen då trubaduren var närvarande.

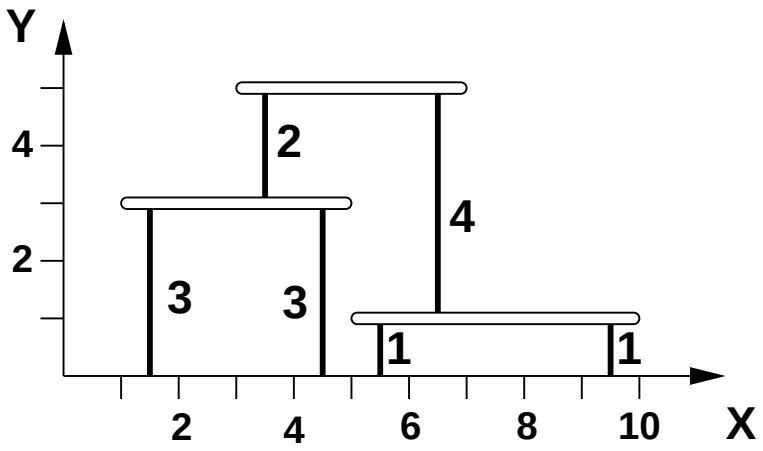
Extra exempel

```
5
7
3 1 2 4
3 3 4 5
4 1 2 3 4
4 1 3 4 5
2 3 4
4 2 3 4 5
3 1 3 5
```

Svar

```
1 3 5
```

UPPGIFT 2 – PLATTFORMAR



FIGUR 1. *Pelarnas placering i exemplet. Den totala längden är* $1 + 1 + 3 + 3 + 2 + 4 = 14$.

En bana håller på att designas till ett nytt tvådimensionellt plattformsspel. En plattform motsvaras av en horisontell linje, och placeringar för alla plattformar har redan bestämts. Till skillnad från de flesta andra plattformsspel så kan dessa plattformar inte hänga fritt i luften, utan måste stadgas upp av pelare. Varje plattform måste ha två pelare, en på vardera sida, som sträcker sig ända ner till marken, eller tills den slår i en platform under sig (se figur ovan). Pelarna börjar en halv längdenhet in från plattformens ändpunkter. Skriv ett program som beräknar summan av längden på alla pelare. Du kan anta att en plattform har försumbar tjocklek.

Indata

Första raden i filen `pelare.dat` innehåller heltalet N ($1 \leq N \leq 100$), antalet plattformar. Därefter följer N rader innehållande tre heltal, Y, X_1 och X_2, som anger en plattforms koordinater. Alla koordinater är mellan 0 och $10,000$. Dessutom gäller att $X_1 + 1 < X_2$, dvs längden på varje plattform kommer vara minst 2. Inga plattformar kommer att överlappa varandra.

Utdata

Skriv ut summan av pelarnas längder.

Exempel

```
3
1 5 10
5 3 7
3 1 5
```

Svar

```
14
```

UPPGIFT 3 – KÖPA BÖCKER

Du ska köpa N böcker av olika slag (numrerade från 1 till N) och kollar därför runt hos internetbokhandlarna. Varje bok finns hos minst en bokhandel och kan variera i pris mellan de olika bokhandlarna. Dessutom kostar det ett visst belopp i porto att beställa från varje bokhandel, oavsett hur mycket man beställer. Skriv ett program som beräknar det minimala beloppet böckerna kostar dig, inräknat porto. Du kan beställa från hur många bokhandlar som helst.

Indata

Första raden i filen `bok.dat` innehåller två tal: N, antalet böcker du ska köpa ($1 \leq N \leq 100$), och M, antalet bokhandlar ($1 \leq M \leq 15$). Därefter följer en rad med två tal som anger antalet böcker K (av de eftersökta) som finns i första bokhandeln, samt portot för denna bokhandel. Detta följs av K rader innehållande två tal: numret på en bok som finns i bokhandeln och dess pris. Denna information upprepas sedan för återstående bokhandlar. Alla priser och porton anges i hela kronor och överstiger inte $10,000$.

Utdata

Programmet ska skriva ut det minimala beloppet böckerna kostar, inräknat portokostnaden för alla bokhandlar du beställer från.

Exempel

```
7 4
4 9
1 28
6 45
3 49
4 108
7 49
1 26
2 179
3 54
4 99
5 129
6 45
7 244
5 20
7 249
2 184
5 133
4 109
6 42
1 0
6 43
```

Svar

```
822
```

Samma exempel i tabellform med de fyra bokhandlarna A, B, C och D, samt de sju böckerna B1-B7. P:o står för porto. De skuggade fälten markerar dina köp i den optimala lösningen.

A		**B**		**C**		**D**	
P:o	9	P:o	49	P:o	20	P:o	0
B1	28	B1	26	B7	249	B6	43
B6	45	B2	179	B2	184		
B3	49	B3	54	B5	133		
B4	108	B4	99	B4	109		
		B5	129	B6	42		
		B6	45				
		B7	244				

UPPGIFT 4 – VALUTAVÄXLING

Den välkända banken Dumbank har nyligen öppnat en ny tjänst för sina kunder. Tjänsten är en valutaväxlingstjänst där kunden stoppar in sina pengar i kronor, och därefter växlar sina insatta pengar mot andra valutor. Välkänt är att det sällan är lönt att växla sina pengar fram och tillbaka mellan två valutor, men det Dumbank förbiså är att man kan, om man är listig, växla runt sina pengar mellan flera olika valutor och till sist tillbaka till kronor och sluta med mer pengar än man hade från början.

Efter att ha haft sin tjänst i drift ett tag har Dumbank börjat inse att det går att fiffla med deras system och har därför valt att stänga ner tjänsten om D dagar. Dessutom införde de begränsningen att man bara kan växla pengar en gång per dag.

Du har precis hört talas om denna banktjänst och bestämmer dig för att utnyttja den. Tjänsten tillåter växling mellan N olika valutor numrerade från 1 till N där svenska kronor är nummer 1. Du känner till växlingskurserna mellan alla par av valutor, som dessutom är oförändrade under de D dagar som tjänsten har öppet. Växlingskursen $K[i][j]$ är ett reellt tal som anger hur mycket du kommer ha i valuta j om du växlar in en enhet av valuta i. Om vi exempelvis antar att valuta 2 är amerikanska dollar, så skulle $K[2][1]$ vara cirka 7 med dagens penningvärde.

Skriv ett program som beräknar vad du som mest kan tjäna om du sätter in dina besparingar (i svenska kronor) dag 1. Du hinner därmed som mest göra D växlingar. Efter den sista växlingen måste dina pengar vara i svenska kronor igen.

Indata

Första raden i filen `dumbank.dat` innehåller två heltal, N ($2 \leq N \leq 100$) och D ($2 \leq D \leq 100$). Därefter följer N rader med N tal som anger växlingkurserna, där tal j på rad i motsvarar $K[i][j]$. Växlingkurserna är reella tal med en noggranhet på som mest 4 decimaler. Talen i diagonalen, $K[i][i]$, kommer alltid vara 1.

Utdata

Skriv ut den faktor med vilket du som mest kan växla upp dina insatta pengar, med minst fyra decimaler. Små avrundningsfel är tillåtna.

Exempel

```
4 6
1 1.2 0.001 0.001
0.001 1 1.2 0.001
0.001 0.001 1 1.2
1.2 0.001 0.001 1
```

Svar

```
2.0736
```

Extra exempel

```
7 13
1.0000 4.3515 0.4617 0.5947 0.7079 0.6761 1.1220
0.2368 1.0000 0.1082 0.1408 0.1676 0.1617 0.2630
2.2096 9.4259 1.0000 1.3011 1.5488 1.5088 2.4793
1.7497 7.3168 0.7998 1.0000 1.2023 1.1829 1.9055
1.4553 6.0256 0.6457 0.8318 1.0000 0.9645 1.6007
1.5392 6.3096 0.6828 0.8797 1.0682 1.0000 1.6762
0.9002 3.8019 0.4115 0.5301 0.6373 0.6026 1.0000
```

Svar

```
1.41715
```

UPPGIFT 5 – KVADRATER

Pelle ritar ett antal fyllda kvadrater i olika färger i sitt favoritritprogram Paint. Varje ny kvadrat han ritar kan helt eller delvis täcka en tidigare ritad kvadrat. Den nya kvadraten täcker då givetvis helt över de delar av den undre som den överlappar. Det är därför inte säkert att man i efterhand kan veta den exakta storleken på de kvadrater Pelle ritade. Givet hur hans konstverk ser ut efter att alla kvadrater har ritats, bestäm den största storlek som varje kvadrat ursprungligen kan ha haft.

Indata

Första raden i filen `kvadrat.dat` består av två heltal, N ($1 \leq N \leq 1,000$), storleken på ritytan (som också är kvadratisk), och M ($1 \leq M \leq 9$), antalet kvadrater Pelle ritade. Därefter följer N rader som beskriver bilden efter att alla kvadrater har ritats. Varje rad innehåller N tecken, där en punkt motsvarar en pixel som inte blivit täckt av någon kvadrat, och där en siffra motsvarar en pixel som täckts av minst en kvadrat (där siffran motsvarar den senast ritade av dessa). Siffran 1 betecknar den första kvadraten som ritades o.s.v., upp till siffran M.

Utdata

För varje kvadrat som ritats, skriv ut den största storleken den kan tänkas ha haft, talen separerade med blanksteg. Du kan utgå ifrån att bilden i indatat går att skapa genom att rita M kvadrater. Tänk också på att en kvadrat kan ha helt täckts av en eller flera andra kvadrater.

Exempel

```
10 7
..........
.7..1111..
..224441..
..224441..
..224441..
.6666655..
.6666655..
.6666655..
.6666655..
.66666....
```

Svar

```
4 5 5 3 4 5 1
```

Förklaring: Kvadrat 3, som inte syns längre, kan helt gömma sig bakom kvadrat 6. Kvadrat 2 måste ha minst längd 3, men kan ha ända upp till längd 5 om den gömmer sig bakom kvadrat 4, 5 och 6. Dock inte längd 6 eftersom den då skulle behöva sudda ut några pixlar från kvadrat 1.

UPPGIFT 6 – VÄKTAREN

Fysikstudenten Viktor extraknäcker som väktare på vaktbolaget VaktaSmart AB i grannstaden Vaktestad. Under en natt skall Viktor besöka P olika platser. Dessa förbinds av ett antal vägar (V stycken) som Viktor kan gå till fots. Viktor kan börja vaktpasset vid vilken plats som helst, och även sluta var han vill.

Viktor är väldigt begåvad och har uppfunnit en portabel materietransportör (s.k. teleporter), som han givetvis vill använda för att snabbare klara av sitt arbete så han kan åka hem och sova. Teleporten kan användas för att transportera Viktor från en plats till en annan, och under ett nattpass kan han använda den hur många gånger som helst. Transporten sker ögonblickligen och utan nämnvärd energiåtgång. På grund av begränsningar i konstruktionen måste dock Viktor ha varit på platsen han ska teleporteras till tidigare under samma vaktpass. Hjälp Viktor att planera sitt vaktpass genom att skriva ett program som beräknar hur snabbt han kan besöka alla platser, förutsatt att rutten är optimal. Indatat är konstruerat så att det är möjligt att besöka alla platser.

Indata

Första raden i filen `vaktaren.dat` innehåller två heltal, P ($2 \leq P \leq 20,000$) och V ($P - 1 \leq V \leq 100,000$). Därefter följer V rader som beskriver vägarna. De innehåller tre heltal vardera: a_k, b_k och t_k ($1 \leq a_k, b_k \leq P$ samt $0 \leq t_k \leq 100$), där a_k och b_k anger de två platser vägen går mellan, och t_k den tid det tar att gå längs vägen.

Utdata

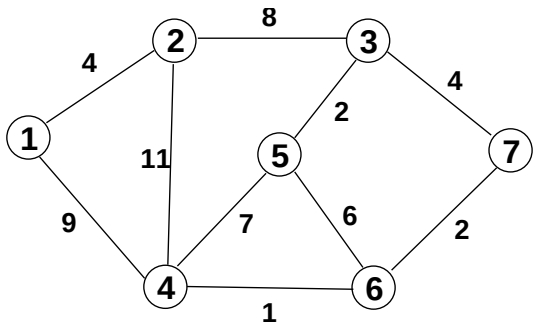
Programmet skall skriva ut den tid som den optimala vaktrundan tar.

Exempel

```
7 10
1 2 4
2 3 8
3 7 4
7 6 2
6 4 1
4 1 9
2 4 11
4 5 7
5 3 2
5 6 6
```

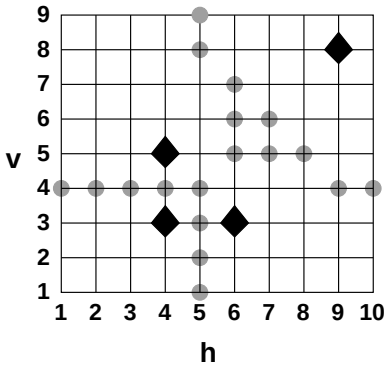
Svar

```
21
```



FIGUR 2. *Kartan med platserna som ska besökas i exemplet. Viktor kan till exempel börja sitt pass på plats 5. Därifrån går han till platserna 3, 2 och 1 (tid 14). Sedan använder han teleportern för att förflytta sig tillbaka till plats 3. Sedan går han till plats 7, 6 och 4 (tid 7). Den totala gångtiden blir 21.*

UPPGIFT 7 – SKYDDSRUM



FIGUR 3. *Kartan i exemplet. De svarta romberna är skyddsrum och de 18 grå cirklarna markerar de korsningar där fler än ett skyddsrum är det närmaste.*

Manhattan är under bombhot. Gamla skyddsrum utspridda på olika ställen i stadsdelen rustas upp och förbereds för användning. När flyglarmen ljuder är det viktigt att alla vet var närmaste skyddsrum är så de kan ta sig dit snabbt. Ett litet problem är dock att vid vissa korsningar är det inte självklart vilket skyddsrum man ska ta sig till; det finns flera att välja på dit avståndet är lika långt (och inget annat är närmare). Skriv ett program som, givet storleken på stadsdelen och var skyddsrummen ligger, bestämmer antalet sådana platser.

Som bekant består gatunätet i Manhattan av horisontella och vertikala gator. För enkelhetens skull betraktar vi stadsdelen som ett rektangulärt rutnät av gator (se figuren ovan). Man kan aldrig snedda utan bara gå på gatorna. Avståndet mellan två korsningar med koordinater (h_1, v_1) och (h_2, v_2) blir därför $|h_1 - h_2| + |v_1 - v_2|$. Skyddsrummen är alltid placerade i en korsning.

Indata

Första raden i filen `newyork.dat` innehåller två heltal H och V ($1 \leq H, V \leq 30,000$), antalet horisontella respektive vertikala gator. Gatorna numreras från 1 till H respektive 1 till V. Därefter följer en rad med talet N ($1 \leq N \leq 10$), antalet skyddsrum. Sedan följer N rader med två heltal vardera, h och v ($1 \leq h \leq H$, $1 \leq v \leq V$), som anger vid vilka korsningar skyddsrummen ligger. Ingen korsning kommer ha mer än ett skyddsrum.

Utdata

Skriv ut ett tal: antalet korsningar där fler än ett skyddsrum är det närmaste.

Exempel

```
10 9
4
4 3
4 5
6 3
9 8
```

Svar

```
18
```