Lëtzebuerger Informatiksolympiad 2014

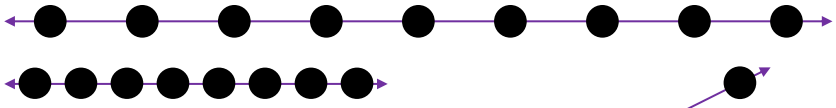
Finale

Problem I (50 Punkte) Der bösartige Frosch

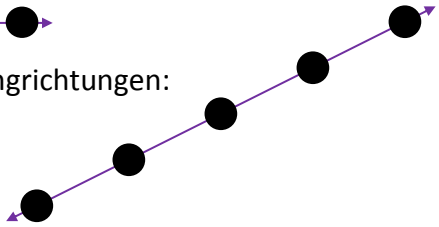
Problemstellung

Die Bösartigkeit des Cheonggaeguri-Frosches ist in Taïwan legendär. Dies ist ein wohlverdienter Ruf, da dieser Frosch nachts durch Reisfelder springt und dabei Reispflanzen umknickt. Morgens, nachdem man festgestellt hat, welche Pflanzen umgeknickt worden sind, möchte man gerne den Weg des Frosches herausfinden, welcher den größten Schaden angerichtet hat. Ein Frosch springt immer geradlinig, mit Sprüngen gleicher Länge, durch ein Reisfeld.

Verschiedene Frösche können verschiedene Sprunglängen haben:



... und verschiedene Sprungrichtungen:



Die Pflanzen eines Reisfeldes befinden sich an den Schnittpunkten eines rechteckigen Gitters (siehe Bild 1). Die Frösche überqueren das Feld immer ganz (siehe Bild 2).

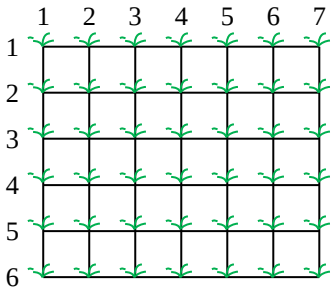


Bild 1

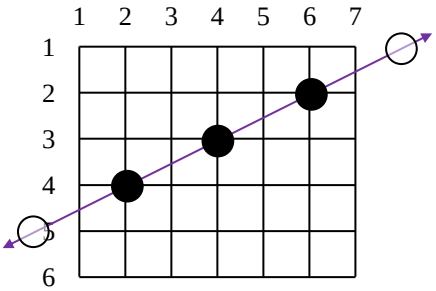


Bild 2

Mehrere Frösche können durch ein Reisfeld springen, von Reispflanze zu Reispflanze. Jeder Sprung endet auf einer Reispflanze, die dabei umgeknickt wird (siehe Bild 3). Im Verlauf einer Nacht können mehrere Frösche auf derselben Pflanze landen. Am nächsten Morgen kann man nur noch

erkennen, welche Pflanzen umgeknickt wurden. Bild 3 zeigt die Wege mehrerer Frösche durch ein Reisfeld und Bild 4 das, was davon am nächsten Morgen noch zu sehen ist.

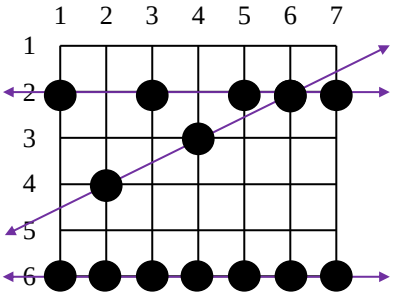


Bild 3

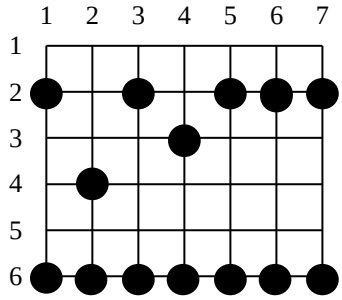


Bild 4

Anhand von Bild 4 kann man alle möglichen Wege, auf denen ein Frosch gehüpft sein könnte, rekonstruieren. Interressant sind nur die Wege, auf denen wenigstens 3 Reispflanzen umgeknickt worden sind. Solch ein Weg wird als Froschweg bezeichnet. In diesem Fall bedeutet das, dass die 3 Wege von Bild 3 alle Froschwege sind (es sind aber auch noch andere Froschwege möglich). Im Gegensatz dazu ist der senkrechte Weg entlang der ersten Spalte kein Froschweg, denn dort sind nur 2 Pflanzen umgeknickt. Der Diagonalweg durch die Schnittpunkte von Reihe 2 und Spalte 3, von Reihe 3 und Spalte 4 und von Reihe 6 und Spalte 7 hat zwar drei umgeknickte Pflanzen, aber keine gleichmäßige Sprunglänge, und ist daher auch kein Froschweg.

Beachte, dass entlang der Linie eines Froschweges mehrere umgeknickte Pflanzen sein können, welche aber nicht zu diesem Weg gehören (siehe die Pflanze (2; 6) auf dem waagerechten Weg entlang der 2. Reihe in Bild 4).

Schreibe ein Programm, das die größte Anzahl an umgeknickten Pflanzen entlang eines Froschweges bestimmt. In Bild 4 hat der Weg durch die sechste Reihe den größten Schaden angerichtet, die Antwort lautet also 7 (umgeknickte Pflanzen).

Programmeingabe

Das Programm liest von der Standardeingabe.

Die erste Zeile enthält 2 ganze Zahlen R (Anzahl der Reihen des Reisfeldes) und S (Anzahl der Spalten des Reisfeldes), $1 \leq R, S \leq 5000$.

Die zweite Zeile enthält eine einzige ganze Zahl N (die Anzahl der umgeknickten Pflanzen), $3 \leq N \leq 5000$.

Jede der restlichen N Zeilen enthält 2 ganze, durch ein Leerzeichen getrennte, Zahlen: die Reihe ($1 \leq$ Reihe $\leq R$) und die Spalte ($1 \leq$ Spalte $\leq S$) einer umgeknickten Pflanze. Jede umgeknickte Pflanze ist nur einmal aufgeführt.

Programmausgabe

Das Programm schreibt in die Standardausgabe. Die Ausgabe enthält nur eine einzige Zeile mit einer einzigen ganzen Zahl: die Anzahl an umgeknickten Pflanzen entlang eines Froschweges, der den größten Schaden angerichtet hat. Ist kein Froschweg vorhanden, dann ist diese Zahl 0.

Beispiele von Eingabe und Ausgabe

Beispiel 1 (das Beispiel von Bild 4)

Eingabe

```
6 7
14
2 1
6 6
4 2
2 5
2 6
2 7
3 4
6 1
6 2
2 3
6 3
6 4
6 5
6 7
```

Ausgabe

```
7
```

Beispiel 2 (das Beispiel von Bild 5)

Eingabe

```
6 7
18
1 1
6 2
3 5
1 5
4 7
1 2
1 4
1 6
1 7
2 1
2 3
2 6
4 2
4 4
4 5
5 4
5 5
6 6
```

Ausgabe

```
4
```

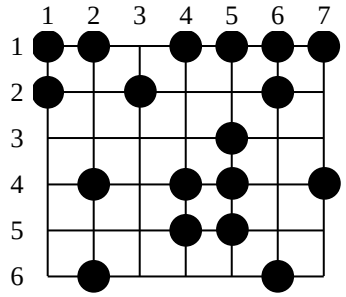


Bild 5

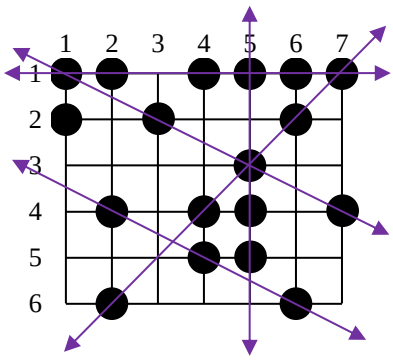


Bild 6: Die grösste Anzahl an umgeknickten Pflanzen ist 4.

Reicht das Quellprogramm unter dem Namen FROSCH.xxx ein, mit xxx=PAS oder C(PP) oder JAVA.

Reicht auch das ausführbare Programm unter dem Namen FROSCH.EXE ein.

Problem II (50 Punkte) Batch Zeiteinteilung

Problemstellung

Gegeben ist eine Folge von N Aufträgen (Jobs) für einen Rechner. Die Aufträge sind von 1 bis N durchnummeriert, so dass die Auftragsfolge 1, 2, …, N lautet. Die Folge von Aufträgen muss in einen oder mehrere Batches aufgeteilt werden, wobei jeder Batch aus einer Folge von fortlaufenden Aufträgen besteht, d. h. die Jobs dürfen nicht umgereiht sondern nur gruppiert werden. Die Ausführung startet zum Zeitpunkt 0. Die Batches werden der Reihe nach folgendermaßen abgearbeitet: Wenn ein Batch b Aufträge mit kleineren Nummern als der Batch c enthält, dann wird Batch b vor Batch c abgearbeitet. Die Aufträge in einem Batch werden hintereinander am Rechner ausgeführt. Unmittelbar nachdem alle Aufträge in einem Batch ausgeführt wurden, gibt der Rechner von allen Aufträgen des Batches die Resultate aus. Die Ausgabezeit von einem Auftrag j ist die Beendigungszeit jenes Batches, der den Auftrag j enthält.

Der Rechner benötigt für jeden Batch eine Anlaufzeit S. Für jeden Job i kennen wir seinen Kostenfaktor F_i und die benötigte Rechenzeit T_i. Wenn ein Batch die Aufträge x, x+1, ..., x+k enthält und zum Zeitpunkt t startet, dann ist die Ausgabezeit von allen Aufträgen in diesem Batch: $t + S + (T_x + T_{x+1} + ... + T_{x+k})$. Beachte, dass der Rechner die Ergebnisse aller Aufträge eines Batches zur gleichen Zeit ausgibt. Wenn die Ausgabezeit eines Auftrages i gleich O_i ist, dann betragen seine Kosten $O_i \times F_i$.

Nehmen wir 5 Aufträge an, die Anlaufzeit $S = 1$, $(T_1, T_2, T_3, T_4, T_5) = (1, 3, 4, 2, 1)$ und $(F_1, F_2, F_3, F_4, F_5) = (3, 2, 3, 3, 4)$. Wenn die Aufträge in drei Batches {1, 2}, {3}, {4,5} aufgeteilt werden, dann betragen die Ausgabezeiten $(O_1, O_2, O_3, O_4, O_5) = (5, 5, 10, 14, 14)$ und die Kosten der Aufträge jeweils (15, 10, 30, 42, 56). Die Gesamtkosten für eine Aufteilung ist die Summe der Kosten aller Aufträge. Somit betragen die Gesamtkosten für unser Beispiel von oben 153.

Du sollst ein Programm schreiben, welches bei gegebener Batch-Anlaufzeit und einer Folge von Aufträgen mit ihren Ausführungszeiten und Kostenfaktoren die minimal möglichen Geamtkosten berechnet.

Programmeingabe

Dein Programm soll von der Standardeingabe lesen. Die erste Zeile enthält die Anzahl N der Aufträge, $1 \le N \le 10000$. Die zweite Zeile enthält die Batch-Anlaufzeit S. S ist ganzzahlig und es gilt: $0 \le S \le 50$. Die folgenden N Zeilen enthalten Informationen über die Aufträge 1, 2, …, N im nachstehenden Format. Zuerst in jeder Zeile steht eine ganze Zahl T_i, $1 \le T_i \le 100$, die Rechenzeit

für einen Auftrag. Anschließend die ganze Zahl F_i, $1 \le F_i \le 100$, der Kostenfaktor für den Auftrag.

Programmausgabe

Dein Programm schreibt in die Standardausgabe. Die Ausgabe enthält eine Zeile mit einer ganzen Zahl: Die minimal möglichen Gesamtkosten.

Beispiele von Programmeingabe und Programmausgabe

Beispiel 1

Eingabe

```
2
50
100 100
100 100
```

Ausgabe

```
45000
```

Beispiel 2 (das Beispiel im Text)

Eingabe

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

Ausgabe

```
153
```

Bemerkung

Für jeden Testfall übersteigen die Gesamtkosten für jede Unterteilung $2^{31} - 1$ nicht.

Reicht das Quellprogramm unter dem Namen BATCH.xxx ein, mit xxx=PAS oder C(PP) oder JAVA.

Reicht auch das ausführbare Programm unter dem Namen BATCH.EXE ein.