

CEOI 2026



Tag 2
(Deutsch (D))

Day 2
(German (D))

Blumenschneiden

Zeitlimit: 4 s Speicherlimit: 256 MiB

Im CEOI-Gemeinschaftsgarten züchten wir eine Sammlung besonderer Blumen, die ihre Wurzeln eng miteinander verflechten. Wenn wir sie durchtrennen, wachsen sie wieder zusammen, solange wir nicht zu aggressiv schneiden und irreparablen Schaden anrichten.

Wenn ein Blumenpaar, a und b , seine Wurzeln miteinander verflechtet, nennen wir die beiden Blumen „verbunden“. Andernfalls sind sie „getrennt“.

Die Wurzeln wachsen nach folgender Regel: Seien a und b zwei getrennte Blumen. Wenn mindestens zwei weitere Blumen c und d existieren, sodass sowohl a als auch b mit c und mit d verbunden sind, dann wachsen Wurzeln zwischen a und b , und sie werden verbunden.

Diese Blumen wachsen nun schon seit einiger Zeit, und alle Wurzeln, die gemäß der obigen Regel wachsen konnten, sind bereits gewachsen. Mit anderen Worten: Wenn zwei Blumen a und b beide mit gewissen Blumen c und d verbunden sind, dann ist garantiert, dass auch a und b miteinander verbunden sind.

Wir müssen diesen Garten ausgraben und an den nächsten CEOI-Austragungsort verlegen. Um den Umzug zu vereinfachen, möchten wir möglichst viele Wurzeln durchtrennen. Allerdings sollen die Blumen wieder in ihren aktuellen Zustand zusammenwachsen können. Wie groß ist die maximale Anzahl verbundener Blumenpaare, deren Verbindung wir durchtrennen dürfen, sodass sich der ursprüngliche Zustand dennoch wiederherstellen lässt? Dabei spielt es keine Rolle, wie viele Wachstumsiterationen dafür erforderlich sind.

Eingabe

Die erste Zeile der Eingabe enthält zwei durch Leerzeichen getrennte ganze Zahlen n und m , die Anzahl der Blumen bzw. die Anzahl der bestehenden Verbindungen zwischen ihnen. Danach folgen m Zeilen, die jeweils ein Paar ganzer Zahlen a_i und b_i enthalten und angeben, dass die Blumen a_i und b_i verbunden sind.

Die Blumen sind mit den ganzen Zahlen $1 \dots n$ nummeriert.

Es ist garantiert, dass die Eingabe die in der Aufgabenstellung beschriebene Regel erfüllt.

Einschränkungen

- $1 \leq n \leq 1000$
- $1 \leq m \leq 10^5$

Teilaufgaben

- Teilaufgabe 1 (20 Punkte): $n \leq 10$ und $m \leq 20$
- Teilaufgabe 2 (14 Punkte): $m = \frac{n \cdot (n-1)}{2}$
- Teilaufgabe 3 (15 Punkte): Es wird garantiert, dass jede Blume mit höchstens 7 anderen Blumen verbunden ist.
- Teilaufgabe 4 (15 Punkte): $n \leq 50$ und $m \leq 1000$
- Teilaufgabe 5 (36 Punkte): Keine zusätzlichen Einschränkungen.

Ausgabe

Gib eine einzelne ganze Zahl aus — die maximale Anzahl von Verbindungen, die wir durchtrennen dürfen.

Beispiel

Eingabe

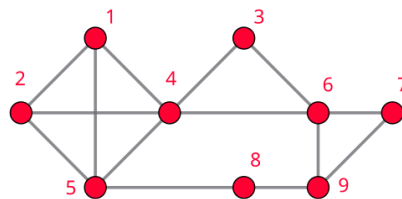
```
9 14
1 2
1 4
1 5
2 4
2 5
3 4
4 5
3 6
4 6
6 7
6 9
7 9
8 9
5 8
```

Ausgabe

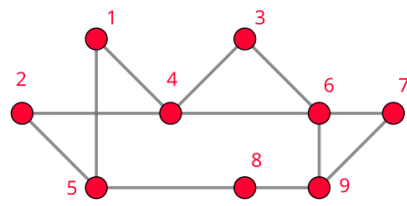
```
2
```

Kommentar

Die Blumen im Beispiel vor dem Durchtrennen irgendwelcher Verbindungen. Man kann überprüfen, dass gemäß dem beschriebenen Wachstumsverfahren keine zusätzlichen Verbindungen entstehen können.



Die Blumen im Beispiel nach dem Durchtrennen besitzen 2 Verbindungen weniger. Die Verbindung zwischen 1 und 2 kann wieder zusammenwachsen, weil die Blumen 1 und 2 beide mit den Blumen 4 und 5 verbunden sind. Ebenso kann die Verbindung zwischen 4 und 5 wieder zusammenwachsen, weil die Blumen 4 und 5 beide mit den Blumen 1 und 2 verbunden sind.



Türme

Zeitlimit: 1 s Speicherlimit: 256 MiB

Du hast n Computer und m Türme, die sich alle an unterschiedlichen Positionen auf einer Geraden befinden. Du musst die Computer mithilfe von Kabeln paarweise verbinden, sodass jedes Kabel bei einem Computer beginnt, einige Türme besucht und bei einem anderen Computer endet. Das Kabel kann jeden beliebigen Turm (nicht nur diejenigen zwischen den Computern) in beliebiger Reihenfolge besuchen. Es kann Türme überspringen, indem es an ihnen vorbeiführt, ohne sie zu besuchen. Es kann auch gar keine Türme besuchen und die beiden Computer direkt verbinden, darf jedoch keinen Computer mit sich selbst verbinden. Die Anzahl der Computer ist gerade.

Seien a und b die Positionen zweier Computer und seien $x_1, \dots, x_k$ die Positionen der Türme, die das Kabel besucht. Die Länge des Kabels ist $|a - x_1| + |x_1 - x_2| + \dots + |x_{k-1} - x_k| + |x_k - b|$. Für ein Kabel definieren wir seinen Wert als $f \cdot u - l$, wobei l seine Länge ist, f eine feste Konstante und u die Anzahl der unterschiedlichen Türme ist, die das Kabel unter $x_1, \dots, x_k$ besucht. Mehrere Kabel können denselben Turm besuchen, und dieser Turm trägt zum Wert jedes dieser Kabel bei.

Du musst die maximal mögliche Summe der Kabelwerte berechnen, wenn alle Computer paarweise verbunden werden (d. h. jeder Computer muss genau zu einem Paar gehören bzw. äquivalent dazu genau mit einem Kabel verbunden sein).

Eingabe

Die erste Zeile enthält die Anzahl der Testfälle T . Die Testfälle folgen nacheinander. Jeder Testfall besteht aus drei Zeilen. Die erste Zeile enthält drei ganze Zahlen n , m und f — die Anzahl der Computer, die Anzahl der Türme und die Konstante f . Die zweite Zeile enthält n ganze Zahlen $a_1, a_2, \dots, a_n$ — die Positionen der Computer. Die dritte Zeile enthält m ganze Zahlen $b_1, b_2, \dots, b_m$ — die Positionen der Türme.

Einschränkungen

Seien N und M die Summen von n bzw. m über alle Testfälle.

- $1 \leq T \leq 10^4$
- $1 \leq N, M \leq 2 \cdot 10^5$
- $0 \leq f \leq 10^9$
- n ist gerade
- $1 \leq a_i, b_i \leq 10^9$
- Alle Positionen der Computer und Türme sind paarweise verschieden (innerhalb eines einzelnen Testfalls).

Teilaufgaben

- Teilaufgabe 1 (5 Punkte): $N \leq 5000$, $m = 1$

- Teilaufgabe 2 (10 Punkte): $T \leq 20$, $n \leq 10$, $m \leq 100$
- Teilaufgabe 3 (27 Punkte): $N, M \leq 5000$
- Teilaufgabe 4 (21 Punkte): $N \leq 5000$
- Teilaufgabe 5 (37 Punkte): Keine zusätzlichen Einschränkungen.

Ausgabe

Gib T ganze Zahlen, jeweils in einer eigenen Zeile, aus — die maximal mögliche Summe der Kabelwerte für jeden Testfall.

Beispiel

Eingabe

```
4
2 1 100
1 10
11
4 1 10
2 4 6 8
20
4 1 10
2 4 6 8
5
6 3 10
2 13 4 8 6 10
5 1 9
```

Ausgabe

```
89
-4
12
51
```

Vim

Zeitlimit: 1 s Speicherlimit: 256 MiB

Es wäre schwer, etwas noch *nerdigeres* als das altmodische Vim zu finden – einen Editor, in dem wir Dinge tun können, von denen wir in „normalen“ Editoren nur träumen können. Vorausgesetzt natürlich, wir sind bereit, eine Woche ins Lernen und einen Monat ins Üben zu investieren, damit die ungewöhnlichen, aber mächtigen Befehle, die es bietet, zu unserem „Muskelgedächtnis“ werden. Nun, wie auch in anderen Editoren haben wir in Vim einen *Cursor*, der sich immer auf einem Zeichen befindet. Anfangs befindet er sich auf dem ersten Zeichen des Textes, und wie auch in anderen Editoren gibt es in Vim eine *Zwischenablage* (Clipboard), die zum Speichern (und indirekt zum Kopieren und Verschieben) von Textstücken gedacht ist. Die Zwischenablage ist anfangs leer.

In dieser Aufgabe nehmen wir an, dass sich zu Beginn genau ein Zeichen '-' (minus) im Editor befindet, und unser Ziel ist es, genau n aufeinanderfolgende Minuszeichen zu erzeugen. Dafür verwenden wir vier Befehle:

- **h**: Befindet sich der Cursor auf dem ersten Zeichen, dann bewirkt dieser Befehl nichts, andernfalls bewegt er den Cursor auf das nächste Zeichen links.
- **l**: Befindet sich der Cursor auf dem letzten Zeichen, dann bewirkt dieser Befehl nichts, andernfalls bewegt er den Cursor auf das nächste Zeichen rechts.
- **Y**: Die Zeichenfolge von der Cursorposition bis zum Ende des Textes wird in die Zwischenablage kopiert und überschreibt dabei deren bisherigen Inhalt. (Befindet sich der Cursor auf dem ersten Zeichen des Textes, wird der gesamte Text in die Zwischenablage kopiert.)
- **P**: Eine Kopie des in der Zwischenablage gespeicherten Textes wird vor dem Zeichen eingefügt, auf dem sich der Cursor befindet, und der Cursor wird auf das letzte eingefügte Zeichen verschoben. Der Inhalt der Zwischenablage wird nicht verändert. Ist die Zwischenablage leer, passiert nichts.

Schreibe ein Programm, das die Zahl n einliest und die minimale Anzahl von Befehlen ausgibt, die benötigt wird, um unter den beschriebenen Bedingungen in Vim mit genau n aufeinanderfolgenden '-'-Zeichen zu enden. Das Programm soll außerdem ein Beispiel für eine Befehlsfolge mit dieser minimalen Anzahl an Befehlen ausgeben.

Eingabe

Die erste Zeile enthält die Anzahl der Testfälle t . Die folgenden t Zeilen enthalten Testfälle mit der gewünschten Zeichenfolgenlänge n .

Einschränkungen

- $1 \leq t \leq 100$
- $1 \leq n \leq 10^7$

Teilaufgaben

Wenn du nur die korrekte Anzahl an Befehlen ausgibst, aber kein Beispiel oder ein inkorrektes Beispiel für eine Befehlsfolge, erhältst du die Hälfte der Punkte für die entsprechende Teilaufgabe.

- Teilaufgabe 1 (20 Punkte): $n \leq 100$
- Teilaufgabe 2 (8 Punkte): $n \leq 1000$
- Teilaufgabe 3 (18 Punkte): $n \leq 10^4$
- Teilaufgabe 4 (18 Punkte): $n \leq 10^5$
- Teilaufgabe 5 (18 Punkte): $n \leq 10^6$
- Teilaufgabe 6 (18 Punkte): Keine weiteren Einschränkungen.

Ausgabe

Gib für jeden Testfall eine Zeile aus, in der die benötigte minimale Anzahl an Befehlen sowie ein Beispiel einer solchen Befehlsfolge steht.

Beispiel

Eingabe

2
21
2

Ausgabe

10 YPYPhPYPPP
2 YP

Kommentar

Wie wir in der folgenden Tabelle sehen können, liefert im ersten Fall die Folge YPYPhPYPPP das korrekte Ergebnis. Das Zeichen '=' in der Spalte Bildschirm stellt das Zeichen '-' dar, auf dem sich der Cursor befindet.

Schritt	Befehl	Bildschirm	Zwischenablage
0		=	(leer)
1	Y	=	-
2	P	=-	-
3	Y	=--	--
4	P	=---	--
5	h	=----	--
6	P	=-----	--
7	Y	=-----	-----
8	P	-----=-----	-----
9	P	-----=-----	-----
10	P	-----=-----	-----