D. Laser Strike

Problemname	Laser Strike
Time Limit	3 Sekunden
Memory Limit	1 Gigabyte

Ann und ihre Freundin Kathrin haben kürzlich ein neues Brettspiel entdeckt, das ihr Lieblingsspiel geworden ist: Laser Strike. In diesem Spiel arbeiten zwei Spielerinnen zusammen, um N Teile vom Brett zu entfernen. Das Spiel hat zwei Phasen. Der Haken ist, dass Kathrin nicht alle Informationen über das Spiel hat. Um das Spiel zu gewinnen, müssen Ann und Kathrin zusammenarbeiten und dabei so wenig wie möglich kommunizieren.

Es gibt N einzigartige Teile auf dem Brett, welche von 0 bis $N-1$ durchnummeriert sind. Beide Spielerinnen können diese Teile sehen. Es gibt $N-1$ Verbindungen zwischen jeweils zwei Teilen, sodass es möglich ist, jedes Teil von jedem anderen Teil mit Hilfe der Verbindungen zu erreichen. In anderen Worten: Die Verbindungen bilden einen Baum. **Nur Ann kann dieses Verbindungen sehen; Kathrin kennt sie nicht.**

In der ersten Phase des Spiels wählt Ann eine Reihenfolge $\ell_0, \ell_1, \ldots, \ell_{N-2}$, in der die Teile entfernt werden sollen, bis ein einziges übrig bleibt. Diese Reihenfolge wird vor Kathrin geheim gehalten. Sie gewinnen das Spiel, wenn Kathrin die Reihenfolge nachbilden kann. Das Entfernen der Teile muss der folgenden Regel folgen: jedes Mal, wenn ein Teil entfernt wird, muss es exakt mit einem vebleibenden Teil verbunden sein. In anderen Worten: Das entfernte Teil muss ein Blatt in dem aus dem Teil selbst und den verbleibenden Teilen bestehenden Baum sein. (Nachdem $N-1$ Teile enfernt wurden, wird das letzte Teil automatisch entfernt und sie gewinnen das Spiel.) Ann muss eine Reihenfolge wählen, die der obigen Regel folgt.

Ann wird Kathrin eine Nachricht in der Form einer binären Zeichenkette aufschreiben. Ann kann wählen wie lang die Nachricht ist – je kürzer sie ist, desto mehr Punkte bekommen sie.

Anschließend startet die zweite Phase des Spiels. Das Ziel des Spiels ist es, dass Kathrin $N-1$ Teile vom Brett in der Reihenfolge $\ell_0, \ell_1, \ldots, \ell_{N-2}$ entfernt. Sie wird $N-1$ Züge machen. Vor dem i-ten Zug sagt Ann Kathrin zwei ganze Zahlen a, b mit den folgenden Eigenschaften:

- $a < b$;
- es gibt immer noch ein Paar von direkt miteinander verbundenen Teilen mit den Nummern a und b; und

- entweder a oder b ist das korrekte Teil ℓ_i das Kathrin in diesem Zug entfernen sollte.

Beachte, dass für Ann die Verbindung (a, b) eindeutig durch das Blatt ℓ_i im aktuellen Baum bestimmt ist.

Kathrin entfernt dann entweder a oder b vom Brett. Wenn das korrekte Teil, also ℓ_i, entfernt wurde, spielen sie weiter. Ansonsten verlieren sie das Spiel.

Deine Aufgabe ist es, sowohl Anns also auch Kathrins Strategie so zu implementieren, sodass sie das Spiel gewinnen.

Dein Programm wird basierend auf der Länge der Nachricht, welche Ann in der ersten Phase des Spiels schreibt, bewertet.

Implementierung

Dies ist ein Mehrphasenproblem. Das heißt, dein Programm wird zweimal ausgeführt. Wenn es zum ersten Mal ausgeführt wird, soll es Anns Strategie für die erste Phase des Spiels umsetzen. Beim zweiten Mal soll Kathrins Strategie für die zweite Phase durchgeführt werden.

Die erste Zeile des Inputs enthält zwei ganze Zahlen, P und N. P ist entweder 1 oder 2 und sagt aus, ob wir uns in der ersten oder in der zweiten Phase befinden. N ist die Anzahl Teile.

Der darauffolgenden Input ist abhängig von der Phase:

Phase 1: Ann

Nach der ersten Zeile (wie oben definiert) beschreiben die nächsten $N - 1$ Zeilen den Baum. Jede Zeile enthält zwei Zahlen, a und b ($0 \leq a < b \leq N - 1$), die eine Verbindung zwischen Teilen a und b darstellen.

Dein Programm soll zuerst einen binären String mit höchstens $1\,000$ Zeichen ausgeben. Jedes Zeichen ist entweder `0` oder `1`. Das ist die Nachricht von Ann. Für einen String der Länge 0 soll eine leere Zeile ausgegeben werden.

Danach sollen $N - 1$ Ganzzahlen $\ell_0, \ell_1, \ldots, \ell_{N-2}$ auf separaten Zeilen ausgegeben werden. Diese stellen die Reihenfolge dar, in der Ann die Blätter des Baums entfernen möchte. Die Reihenfolge muss so sein, dass das entfernte Teil immer ein Blatt ist, wenn die Teile nacheinander entfernt werden. Das heißt, der Baum muss immer verbunden sein.

Phase 2: Kathrin

Nach der ersten Zeile (wie oben definiert) enthält die nächste Zeile des Inputs den binären String, Anns Nachricht der ersten Phase.

Danach gibt es $N-1$ Runden der Interaktion, eine für jede von Kathrins Aktionen.

In der $i$ten Aktion soll dein Programm zwei Zahlen, a und b ($0 \leq a < b \leq N-1$) einlesen. Eines dieser Teile ist das Blatt ℓ_i in Anns Reihenfolge und das andere ist das einzige noch verbleibende und zu ℓ_i verbundene Teil. Dann soll dein Programm ℓ_i ausgeben, was dafür steht, dass Kathrin dieses Blatt entfernt. Fals dein Programm nicht das richtige Blatt ℓ_i ausgibt, verlieren die Mädchen das Spiel und du erhälst den Verdict Wrong Answer für diesen Testfall.

Details

Falls die *Summe* der Laufzeiten der zwei Ausführungen von deinem Programm größer ist als die Zeitlimite, wird deine Submission die Bewertung Time Limit Exceeded bekommen.

Stelle sicher, dass du den Standard Output nach jeder Zeile flushst. Sonst könnte dein Programm Time Limit Exceeded erhalten. In Python passiert dies automatisch, wenn du `input()` verwendest, um Zeilen einzulesen. In C++ leert `cout << endl;` den Ausgabepuffer (flush) zusätzlich zum Ausgeben eines Zeilenumbruchs. Falls du printf benutzt, nutze `fflush(stdout);`.

Bedenke, dass es knifflig sein kann, einen leeren String einzulesen. Das gegebene Template geht korrekt mit diesem Fall um.

Einschränkungen und Bewertung

- $N = 1\,000$.
- $0 \leq a < b \leq N-1$ für alle Verbindungen.

Deine Lösung wird an einer Menge von Testgruppen getestet, die jede eine Anzahl Punkte gibt. Jede Testgruppe enthält eine Reihe an Testfällen. Um die Punkte für eine Testgruppe zu erhalten, musst du alle Testfälle der Gruppe lösen.

Gruppe	Maximale Punktzahl	Einschränkungen
1	8	Der Baum ist ein Stern. Das heißt, alle Knoten außer einem sind Blätter.
2	9	Der Baum ist eine Linie. Das heißt, alle Knoten außer zwei Blätter haben genau zwei benachbarte Knoten.
3	21	Der Baum ist ein Stern mit Linien, die davon abgehen. Das heißt, alle Knoten haben entweder einen oder zwei benachbarte Knoten, außer einer, der mehr als zwei benachbarte Knoten hat.
4	36	Die Distanz zwischen zwei Knoten ist höchstens 10.
5	26	Keine zusätzlichen Einschränkungen.

Für jede Testgruppe, die dein Programm korrekt löst, erhälst du Punkte nach der folgenden Formel:

$$\text{score} = S_g \cdot (1 - 0.3 \cdot \log_{10} \max(K, 1)),$$

wobei S_g die maximale Punktzahl für jede Testgruppe ist. K ist die maximale Länge der Nachricht von Ann über alle Testfälle dieser Testgruppe. **Deine Punktzahl für jede Testgruppe wird auf die nächste ganze Zahl gerundet.**

Die untenstehende Tabelle zeigt die Anzahl Punkte die du erhälst für ein paar ausgewählte Werte von K, wenn dein Programm alle Testgruppen mit diesem K löst. Um 100 Punkte zu erhalten, muss dein Programm jeden Testfall mit $K \leq 1$ lösen.

K	1	5	10	50	100	500	1000
Punktzahl	100	79	70	49	39	20	11

Testtool

Um das Testen deiner Lösung zu vereinfachen, stellen wir ein einfaches Tool zur Verfügung, welches du herunterladen kannst. Siehe dazu bei "attachments" am Ende der Kattis Aufgabenseite nach. Die Benützung des Tools ist freiwillig. Beachte, dass der offizielle Kattis Grader sich vom Testing Tool unterscheidet.

Um dieses Tool zu benutzen, erstelle eine Eingabedatei, z.B. "sample1.in", welches mit einer Zahl N starten sollte, gefolgt von $N-1$ Zeilen, welche den Baum beschreiben. Es sollte dem Format der Phase 1 entsprechen. Zum Beispiel, das folgende Beispiel:

```
7
0 1
1 2
2 3
0 4
0 6
1 5
```

Um Python Programme, z.B. `solution.py` (das normalerweise als `pypy3 solution.py` ausgeführt wird) zu testen, führe den folgenden Befehl aus:

```
python3 testing_tool.py pypy3 solution.py < sample1.in
```

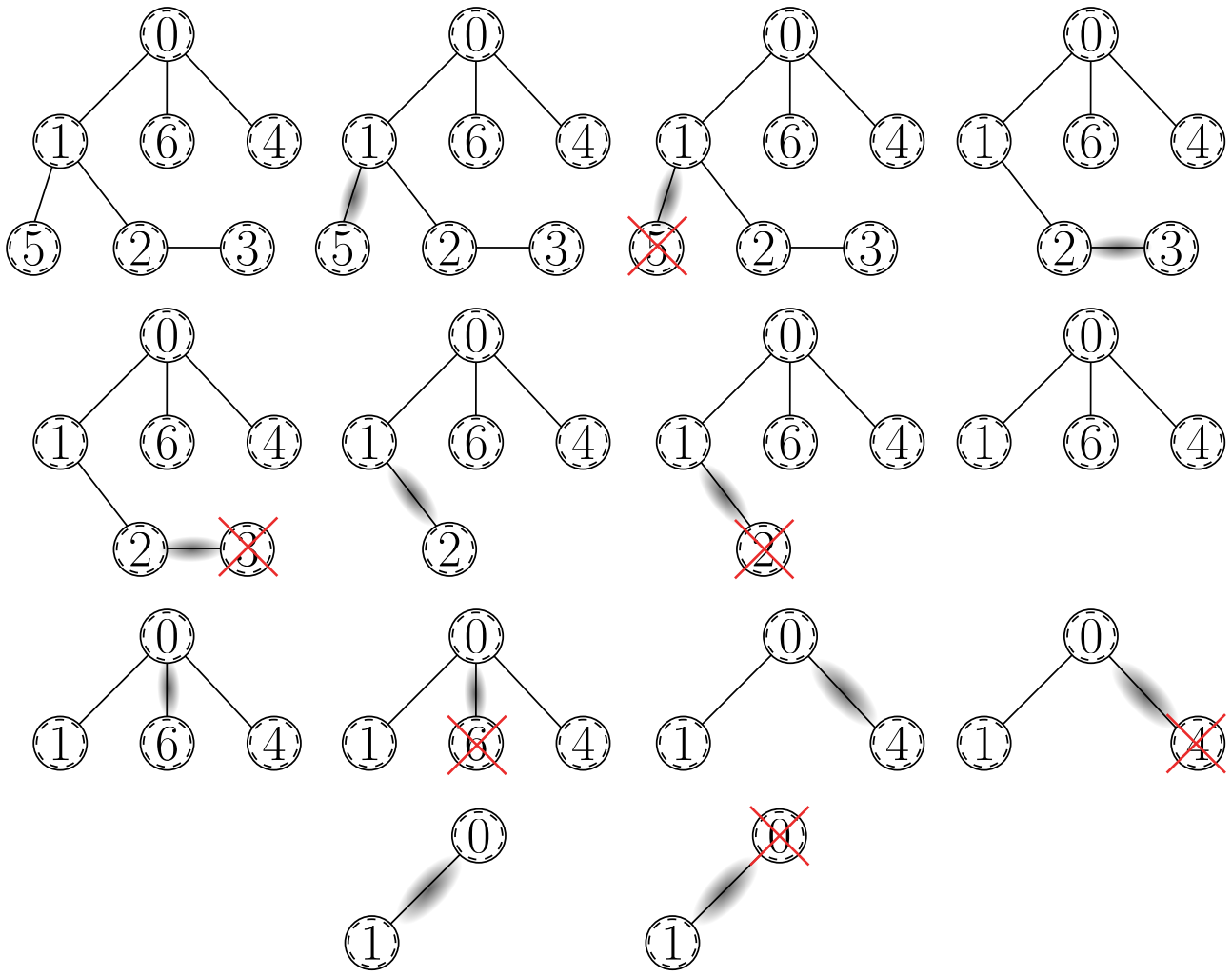
Um C++ Programme zu testen, kompiliere sie zuerst (z.B. mit `g++ -g -O2 -std=gnu++23 -static solution.cpp -o solution.out`) und führe dann den folgenden Befehl aus:

```
python3 testing_tool.py ./solution.out < sample1.in
```

Beispiel

Beachte, dass zur Vereinfachung $N = 7$ für die Beispiele in diesem Abschnitt gilt. Daher sind diese keine validen Testfälle. Es wird nicht erwartet, dass dein Programm diese Fälle lösen kann. Alle Testfälle auf dem Grader werden $N = 1\,000$ haben.

In diesem Beispiel bekommt Ann den folgenden Baum. In der ersten Phase liest Ann den Baum, wählt "`0110`" als binäre Zeichenkette aus und sendet diese an Kathrin. Ann wählt ebenfalls die Reihenfolge $[\ell_0, \ell_1, \ldots \ell_{N-2}] = [5, 3, 2, 6, 4, 0]$, in welcher die Teile aus dem Baum entfernt werden sollten. In der zweiten Phase erhält Kathrin die Zeichenkette "`0110`", welche in der ersten Phase gesendet wurde. Sie bekommt anschließen das Paar $(1, 5)$ und entscheidet, den Knoten 5 zu entfernen, welcher tatsächlich ein Blatt ist. Im nächsten Zug erhält sie das Paar $(2, 3)$ und entfernt Blatt 3. So geht es weiter. Das folgende Bild stellt diese Interaktion dar.



Grader Ausgabe	Deine Ausgabe
1 7	
0 1	
1 2	
2 3	
0 4	
0 6	
1 5	
	0110
	5
	3
	2
	6
	4
	0

Grader Ausgabe	Deine Ausgabe
2 7	
0110	
1 5	
	5
2 3	
	3
1 2	
	2
0 6	
	6
0 4	
	4
0 1	
	0