Weitere Information zu *Tobis Turnier*

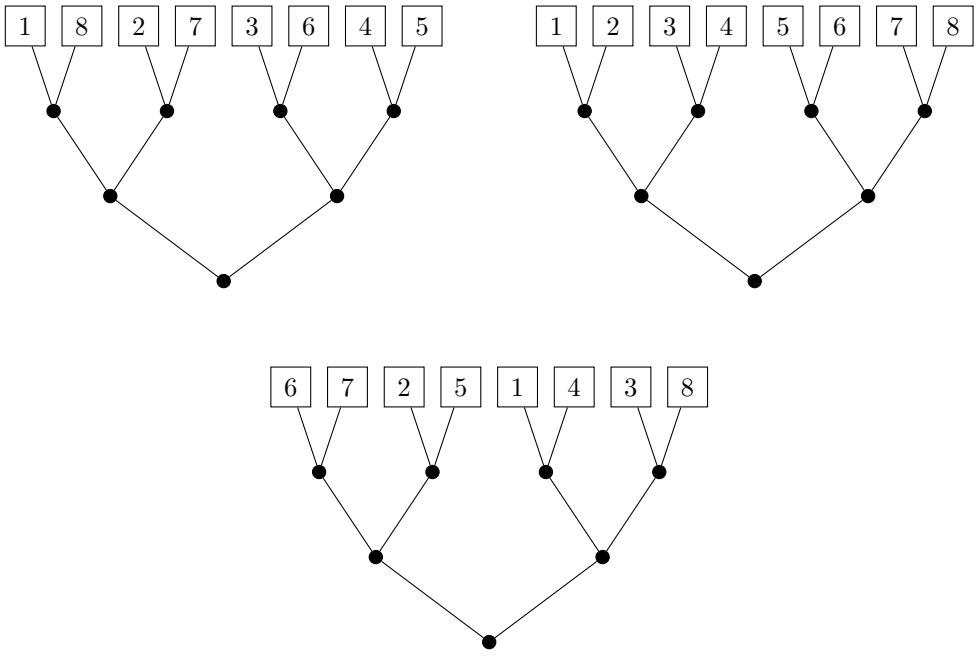
Wir nehmen an, dass die Spieler nummeriert sind. Die Nummern beginnen bei 1 und sind fortlaufend vergeben. Weiterhin nehmen wir an, dass die Anzahl der Spieler eine Zweierpotenz ist. Wir erwarten Ergebnisse mindestens für die Fälle, dass 8 oder 16 Spieler sind wie in unseren Beispieleingaben.

Turnierformen Folgende Turnierformen sollen realisiert werden:

Liga Jeder Spieler spielt einmal gegen jeden anderen Spieler. Der Spieler mit den meisten Siegen gewinnt. Bei Gleichstand gewinnt unter den Spielern mit der höchsten Anzahl von Siegen der mit der kleinsten Spielernummer.

K.O. Die Spieler werden gemäß eines Turnierplans in Paare eingeteilt, so dass kein Spieler übrig bleibt. Die beiden Spieler jedes Paars spielen einmal gegeneinander. Der siegreiche Spieler kommt in die nächste Runde. Das Verfahren wird mit der siegreichen Hälfte der Spieler fortgeführt. Das wird wiederholt, bis nur ein Spieler übrig ist. Dieser gewinnt.

Ein konkreter Turnierplan kann zum Beispiel folgende Formen annehmen:



Teste verschiedene solche konkrete Turnierpläne.

K.O.×5 Diese Turnierform verwendet denselben Ablauf wie K.O. Anstatt nur einmal gegeneinander zu spielen, spielen die Spieler eines Paares nun fünfmal gegeneinander, und der Spieler mit den meisten Siegen kommt in die nächste Runde.

Spielstärken Ermittle Ergebnisse mindestens für die folgenden vier Spielstärken der TeilnehmerInnen. Jede Datei enthält eine Zeile pro TeilnehmerInnen. In der i. Zeile steht die Spielstärke von Spieler i, durchnummeriert ab 1.

```
spielstarke1.txt
spielstarke2.txt
spielstarke3.txt
spielstarke4.txt
```

Zufallszahlengeneratoren Es gibt in jeder gängigen größeren Programmiersprache Unterstützung zur Generierung von (Pseudo)zufallszahlen. Ein typischer Zufallszahlengenerator liefert bei jedem Aufruf eine Zahl zwischen 0 und 1. In Python erhält man eine Zufallszahl zwischen 0 und 1 auf folgende Weise:

```
from random import *
r = random()
```

In Java erhält man eine Zufallszahl zwischen 0 und 1 folgendermaßen:

```
import java.lang.Math;
class RNG {
    public static void main(String args[])
    {
        double r = Math.random();
    }
}
```

In C++11 erhält man eine Zufallszahl zwischen 0 und 1 zum Beispiel so:

```
#include <random>
using namespace std;
int main()
{
    mt19937 mt_rand(time(0));
    double r = mt_rand() / (double)mt_rand.max();
    return 0;
}
```

Wir beschreiben kurz an einem Beispiel, wie man dies nutzen kann, um das in der Aufgabe erklärte Urnenmodell zu realisieren. Nehmen wir an, dass wir ein Spiel zwischen Spieler 1 mit Spielstärke 42 und Spieler 2 mit Spielstärke 77 realisieren wollen. Also gibt es in der Urne 42 Kugeln von Spieler 1, 77 Kugeln von Spieler 2 und 119 Kugeln insgesamt.

Wir verwenden unseren Zufallszahlengenerator und erhalten eine Zahl r, die mindestens 0 und höchstens 1 ist. Wenn r zwischen 0 und $42/119$ liegt, dann erklären wir Spieler 1 zum Sieger, ist r mindestens $42/119$, dann erklären wir Spieler 2 zum Sieger.