1/2

Aufgabe 3: Tobis Turnier

Team-ID: ?????

Team: Teamname (oder dein Name)

Bearbeiter/-innen dieser Aufgabe:  
Christopher Besch, Katharina Libner

10. November 2020

[Lösungsidee 1](#_Toc55923576)

[Unterüberschriften 2](#_Toc55923577)

[Umsetzung 2](#_Toc55923578)

[Beispiele 3](#_Toc55923579)

[Quellcode 3](#_Toc55923580)

Anleitung: Trage oben die Aufgabennummer, die Team-ID, den Team-Namen und alle Bearbeiter/-innen dieser Aufgabe mit Vor- und Namen ein. Vergiss nicht, auch den Aufgabennamen anzupassen (statt „LibreOffice-Dokument“)!

Drücke dann F9 zur Aktualisierung, lösche diese Anleitung und lege los :)

Wenn du neue Überschriften einfügst, solltest du das Inhaltsverzeichnis mittels Rechtsklick aktualisieren.

**Die gelb hinterlegten Texte, die hier bereits stehen, geben ein paar Hinweise zur Einsendung. Du solltest sie aber in deiner Einsendung wieder entfernen!**

# Lösungsidee

Es soll die Turniervariante gefunden werden, bei dem der Spieler mit der höchsten Spielstärke am häufigsten gewonnen hat.

Die verschiedenen Turniervarianten werden alle jeweils n-mal simuliert und anschließend miteinander verglichen.

Beim Vergleichen kommt es auf die gewonnenen Spiele des Spielers mit der höchsten Spielstärke an. Jeder Sieg dieses Spielers muss also festgehalten werden. Das Spiel mit den meisten Siegen des Spielers mit der höchsten Spielstärke soll ausgegeben werden.

Die Idee der Lösung sollte hieraus vollkommen ersichtlich werden, ohne dass auf die eigentliche Implementierung Bezug genommen wird.

## Unterüberschriften

Wenn nötig, füge sinnvoll formatierte Unterpunkte ein, indem du die Formatvorlage „Überschrift 2“ (oder „Heading 2“) anwendest. Vergiss nicht, das Inhaltsverzeichnis anschließend wieder mittels Rechtsklick zu aktualisieren.

# Umsetzung

Die Lösungsidee wird in Python implementiert. Das gesamte Programm ist in verschiedene Pythondatein unterteilt, damit die Zusammenhänge der Methoden deutlicher erscheinen.

Die **Main-Methode** führt das Programm aus und läuft wie folgt ab:

Zunächst wird beim Starten des Programms vom Ausführer verlangt die Anzahl der Wiederholungen und die Testdatei der Spielstärken in der Konsole als Argumente einzugeben.

Das Programm liest mit Hilfe der Methode **read\_skill\_levels** die Spielstärken chronologisch ein und gibt sie in Form einer Liste zurück. Die Spielstärken stehen in den Testdateien und belegen jeweils eine Zeile.

Darauffolgend werden die verschiedenen Turniervarianten je nach Eingabe der Wiederholungen in einer while-Schleife mehrmals simuliert und die Siege des besten Spielers pro Variante gezählt. Damit nicht jedes Turnier in der gleichen Reinfolge der Spieler abläuft, die gegeneinander antreten, werden pro Wiederholungen, die Reinfolgen der Spieler mit ihren Spielstärken per Zufall neu gemischt.

Es existieren drei Spielvarianten:

1. RNG Liga
2. RNG KO
3. RNG KO5

Die Ausführung der Spielvarianten konnte wie folgt strukturiert werden:

1. Es wird nach dem Gewinner gesucht, dabei müssen folgende Unterschiede beachten werden:

Beim Turnierablauf unterscheidet sich RNG Liga zu RNG KO und RNG KO5. Es werden zwei verschiedene Methoden für die Turnierabläufe, die den Gewinner zurückgeben, implementiert:

1. **find\_winning\_player\_liga**

Diese Methode wird für die 1. Spielvariante benutzt.

Jeder spielt gegen jeden einmal **play\_RNG** und die Siege jedes Spielers werden gezählt. Das ganze Turnier muss simuliert werden, bis der endgültige Gewinner feststeht.

1. **find\_winning\_player\_KO**

Diese Methode wird für die 2. und 3. Spielvariante benutzt.

Alle Spieler werden in Paare aufgeteilt und spielen **play\_RNG** oder **play\_RNG\_5** gegeneinander, die Gewinner werden wiederum in Paare aufgeteilt usw. Das Ganze wurde rekursiv implementiert. Sobald der beste Spieler während des Turniers einmal verliert, ist er aus dem Rennen und die Simulation kann abgebrochen werden. Dann wird ein Wert zurückgegeben, der ungleich der Nummer des besten Spielers ist.

1. Um den Gewinner zu finden, muss das Spiel RNG simuliert werden:

Die erste und zweite Variante unterscheiden sich zur 3. in den Wiederholungen des Spiels RNG. Es werden zwei verschiedene Methoden der Spielmodi implementiert.

1. **play\_RNG**

RNG wird bei der 1. und 2. Variante einmal gespielt.

1. **play\_RNG\_5**

RNG wird bei der 3. Variante fünf mal gespielt.

1. Es wird überprüft, ob der beste Spieler gewonnen hat.

Nachdem der Gewinner ermittelt wurde, wird mit Hilfe der Methode **does\_player\_win** bei jeder Simulation festgestellt, ob es sich um den besten Spieler handelt. Ist dies der Fall wird die Anzahl der Siege des Spiels hochgezählt.

In der **Main-Methode** werden die Siege der Turniervariante des besten Spielers durch Fallunterscheidungen verglichen. Die Spielvariante mit den meisten Siegen wird vom Programm zurückgegeben.

Hier wird kurz erläutert, wie die Lösungsidee im Programm tatsächlich umgesetzt wurde. Hier können auch Implementierungsdetails erwähnt werden.

# Beispiele



Genügend Beispiele einbinden! Die Beispiele von den BWINF-Webseiten sollten hier diskutiert werden, aber auch eigene Beispiele sind sehr gut – besonders wenn sie Spezialfälle abdecken. Bitte jedoch nicht 30 Seiten Programmausgabe hier einfügen!

# Quellcode

Unwichtige Teile des Programms sollen hier nicht abgedruckt werden. Dieser Teil sollte nicht mehr als 2–3 Seiten umfassen, maximal 10.