Aufgabe 2: Tobis Turnier

Team-ID: 00372

Team-Name: Der Teufel schreibt Java

Bearbeiter/-innen dieser Aufgabe:
Frederico Aberle

23. November 2020

Inhaltsverzeichnis

1 Lösungsidee

Wie die Aufgabenstellung schon sagt, soll das Turnier über viele Wiederholungen simuliert werden. Wenn dabei der spielstärkste Spieler in einer Wiederholung das Turnier gewinnt, erhöht man den Zähler um eins. Am Ende berechnet man die Wahrscheinlichkeit, indem man den Zähler durch die Anzahl der Wiederholungen teilt. Die größte Wahrscheinlichkeit der drei Turniervarianten ist dann unsere Empfehlung.

2 Umsetzung

Im Folgenden werden die beiden Funktionen winner() und winnerBestOfFive näher erläutert.

winner() ist eine Funktion, die zwei Spielstärken als Eingabe nimmt und ausgibt, ob Spieler 1 oder Spieler 2 gewinnt. Dazu generiert es eine zufällige Zahl zwischen 0 und 1. Wenn die erste Spielstärke dividiert durch die Summe der beiden Spielstärke kleiner ist als die generierte Zahl, dann hat Spieler 1 gewonnen und es soll True ausgegeben werden. Ist das Ergebnis größer hat Spieler 2 gewonnen und es soll False ausgegeben werden.

winnerBestOfFive() gibt auch den Gewinner zweier Spieler aus mit dem selben Prinzip wie in winner(), nur mit dem Unterschied, dass sie fünfmal gegeneinander antreten. Wenn also nach fünf Spielen Spieler 1 mehr Siege hat als Spieler 2 soll True ausgegeben werden, ansonsten False.

Alle Turniervarianten werden wiederholt.

Für eine Wiederholung der Liga iteriert man in der ersten for-Schleife durch die Spielstärken und in der zweiten for-Schleife durch die darauffolgende Spielstärke der ersten Schleife bis zur letzten Spielstärke. So erreicht man, dass jeder gegen jeden spielt. Dann ermittelt man mit der Funktion winner() den Gewinner aus den beiden Spielstärken und erhöht den Zähler um 1. Nach den beiden Schleifen ermittelt man den größten Zähler, also die Spielstärke, die am meisten gewonnen hat. Wenn diese Spielstärke gleich dem Spielstärksten entspricht, erhöhen wir einen Zähler für den Spielstärksten um 1.

Für eine Wiederholung des K.O. erstellt man zunächst eine Kopie der Spielstärken, deren Elemente gemischt werden (da verschiedene Turnierpläne getestet werden sollen) und eine Liste mit Indexen von 1 bis n. Das limit der while-Schleife ist zunächst n. Wenn limit kleiner ist als 2 , wird die Schleife verlassen,

da zu diesem Zeitpunkt der Sieger des K.O. Systems feststeht. Innerhalb der while-Schleife iteriert man in einer for-Schleife immer bis limit//2 die Elemente der Kopie durch. Jedes mal stellt man den winner() von der 1. und 2., der 2. und 3., der 3. und 4. Spielstärke, etc. fest und entfernt den Verlierer aus der Kopie. Man hat die Liste der Indexe erstellt, da die Spielstärken beim Löschen neue Indexe annehmen. Daher löscht man noch neben die Spielstärke auch jedes mal den Index der Spielstärke. Nach jeder for-Schleife wird limit halbiert, da wir nach jeder Runde nur noch halb so viele Spielstärken haben. Wenn der Gewinner gleich der Spielstärkste ist, wird - wie bei der Liga - ein Zähler für den Spielstärksten im K.O. System um 1 erhöht.

Für eine Wiederholung des K.O.×5 ist das Prinzip das gleiche wie vom K.O. System. Statt der winner() Funktion ruft man hier jedoch die winnerBestOfFive() Funktion auf.

Die Wahrscheinlichkeiten ergibt sich - wie in der Lösungsidee beschrieben - aus den Zählern der jeweiligen Turniervariante dividiert durch die Anzahl der Wiederholungen der drei Turniervarianten. Zum Schluss wird die größte der drei Gewinn-Wahrscheinlichkeiten für den Spielstärksten als Empfehlung der Turniervarianten ausgegeben.

3 Beispiele

Die eigenen Beispiele beginnen ab spielstaerken5.txt

spielstaerken1.txt
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.3403
Gewinnwahrscheinlichkeit K.O.: 0.1296
Gewinnwahrscheinlichkeit K.O.×5: 0.232
Liga ist die geeignetste Turniervariante

spielstaerken2.txt
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.209
Gewinnwahrscheinlichkeit K.O.: 0.1237
Gewinnwahrscheinlichkeit K.O.×5: 0.2317
K.O.×5 ist die geeignetste Turniervariante

spielstaerken3.txt
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.3201
Gewinnwahrscheinlichkeit K.O.: 0.0624
Gewinnwahrscheinlichkeit K.O.×5: 0.0441
Liga ist die geeignetste Turniervariante

spielstaerken4.txt
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.1161
Gewinnwahrscheinlichkeit K.O.: 0.0612
Gewinnwahrscheinlichkeit K.O.×5: 0.0103
Liga ist die geeignetste Turniervariante

spielstaerken5.txt
Eingabe:
8
100
100
100
100
100
100
100
100
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.1979
Gewinnwahrscheinlichkeit K.O.: 0.1233
Gewinnwahrscheinlichkeit K.O.×5: 0.0313
Liga ist die geeignetste Turniervariante

spielstaerken6.txt
Eingabe:
8
100
1
1
1
1
1
1
1
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.9981
Gewinnwahrscheinlichkeit K.O.: 0.1201
Gewinnwahrscheinlichkeit K.O.×5: 0.1208
Liga ist die geeignetste Turniervariante
spielstaerken7.txt
Eingabe:
16
100
100
100
100
100
100
100
100
100
100
100
100
100
100
100
Ausgabe:
Gewinnwahrscheinlichkeit Liga: 0.1001
Gewinnwahrscheinlichkeit K.O.: 0.0602
Gewinnwahrscheinlichkeit K.O.×5: 0.0094
Liga ist die geeignetste Turniervariante
Empfehlung:
Aufgrund der Testergebnisse empfehle ich die Liga als beste Turniervariante zu wählen.

4 Quellcode

```
1
"""legt den Gewinner nach dem Urnenmodell fest"""
3 def winner(playingStrength1,playingStrength2):
    r = random()
5     if r <= playingStrength1/(playingStrength1+playingStrength2):
        return True
7     if r > playingStrength1/(playingStrength1+playingStrength2):
        return False
9

11 """legt den Gewinner nach fuenfmaligem Wiederholen des Urnenmodells fest"""
def winnerBestOfFive(playingStrength1,playingStrength2):
13     counter1 = 0
    counter2 = 0
15     for b in range(1,5):
        r = random()
```

```
        if r <= playingStrength1/(playingStrength1+playingStrength2):
            counter1 += 1
        if r > playingStrength1/(playingStrength1+playingStrength2):
            counter2 += 1
    if counter1 > counter2:
        return True
    else:
        return False


"""Liga"""
    counter = [0]*n
    for e in range(0,n-1):
        for f in range(e+1,n):
            if winner(strengths[e],strengths[f]) == True:
                counter[e] += 1
            else:
                counter[f] += 1
    largestNumber = max(counter)
    winnerLiga = counter.index(largestNumber)+1
    if winnerLiga == strongest:
        numWinnerLiga += 1


"""K.O."""
    strengthsCopy = copy.copy(strengths)
    shuffle(strengthsCopy)
    indexes = list(range(n))
    limit = n
    while limit >= 2:
        for g in range(0,limit//2):
            if winner(strengthsCopy[g],strengthsCopy[g+1]) == True:
                del strengthsCopy[g+1]
                del indexes[g+1]
            else:
                del strengthsCopy[g]
                del indexes[g]
        limit = limit//2
    winnerKO = indexes[0]+1
    if winnerKO == strongest:
        numWinnerKO += 1
```