Dokumentation Bundeswettbewerbs Informatik 2020

Aufgabe 3: Tobis Turnier

Jakob Paridon

Hinweise:

1. *Im Code habe keine Umlaute (sowie kein „ß“) verwendet, um unnötige Fehlerquellen zu vermeiden.*
2. *Manchmal werden Werte erst dann ausgegeben, nachdem 1 addiert wurde. Dies liegt daran, dass ein Computer beginnt, bei 0 zu zählen, aber ein Mensch normalerweise bei 1.*
3. *Variablen sind dunkelgrau markiert, um sie vom restlichen Text unterscheiden zu können.*

Liga:

Lösungsidee:

Zuerst werden die Anzahl der Spieler, die Stärke der Spieler und die Wiederholungen des Turniers eingelesen. Als nächstes muss ermittelt werden, welcher bzw. welche Spieler die höchste Spielstärke haben.
Als nächstes werden so viele Turniere durchgeführt, wie angegeben wurde. Dies geschieht nach dem System, welches auch im Material vorgestellt wurde. So wird eine Zufallszahl zwischen 0 und 1 generiert und dann mit einer anderen Zahl abgeglichen. Diese Zahl ist die Spielstärke von einem Spieler geteilt durch die Spielstärke beider Spieler addiert. Wenn die Zufallszahl kleiner ist als die errechnete Zahl, gewinnt der Spieler, dessen Zahl geteilt wurde. Sonst gewinnt der andere Spieler.

Umsetzung:

Zur Umsetzung der Idee nutze ich Java in der Version 1.8.
Man kann sich zuerst die Turnierarten aussuchen, dann gibt der Benutzer die Anzahl der Spieler, die Stärke der Spieler und die Wiederholungen des Turniers ein.
Hier sind alle Variablen, die sich nicht selbst erklären, aufgelistet und erklärt:

1. siege[] ist die Anzahl der Siege eines Spielers im Turnier (im Gegensatz zu siegeGesamt[], s. unten)
2. zufall ist eine Variable, die später als Zufallszahl (s. Lösungsidee) genutzt werden wird.
3. spielStaerkeGesamt ist eine Variable, die aus den Spielstärken der beiden Spieler besteht, welche gegeneinander spielen und wird später definiert.
4. staerksterSpieler[] ist eine Variable zum Speichern des/der besten Spieler.

5. anzahlGroessteStaerke ist eine Variable, die die Anzahl der Spieler, welche die höchste Stärke haben, speichert.
6. staerkeGeteilt ist eine Variable, welche den vorhin erklärten Wert, der beim Teilen der Spielstärke eines Spielers durch die Spielstärke beider Spieler addiert, entsteht, speichert.
7. siegeGesamt[] ist die Anzahl der gewonnen Turniere eines jeden Spielers
8. siegeVergleich hält die Anzahl der Siege des Spielers mit den meisten Siegen fest, was später noch wichtig sein wird
9. meisteSiege speichert die Nummer des Spielers mit den meisten Siegen in einem Turnier.

```
        for (int i = 0; i < spielerAnzahl; ++i)
        {
                // groesste Spielstaerke wird festgestellt
                if (spielStaerke[i] > groessteSpielStaerke)
                {
                        groessteSpielStaerke = spielStaerke[i];
                }
        }

        for (int i = 0; i < spielerAnzahl; ++i)
        {
                if (spielStaerke[i] == groessteSpielStaerke)
                {
                        staerksterSpieler[anzahlGroessteStaerke] = i;
                        ++anzahlGroessteStaerke;
                }
        }
```

Q1: Spielstärken der verschiedenen Spieler werden verglichen.

Danach werden die Turniere durchgeführt. Alles wird so oft durchgeführt, wie der Nutzer festgelegt hat.

```
// Die Turniere werden durchgefuehrt
        for (int a = 0; a < wiederholungen; ++a)
        {
                //einige Variablen (ggf. aller Spieler) werden für jedes Turnier auf 0 gesetzt
                for (int i = 0; i < spielerAnzahl; ++i)
                        siege[i] = 0;

                siegeVergleich = 0;
                meisteSiege = 0;

                // ein Turnier wird durchgeführt
                for (int j = 0; j < (spielerAnzahl); ++j)
                {
                        for (int k = 0; k < (spielerAnzahl); ++k)
                        {
                                if (j != k)
                                {
                                        spielStaerkeGesamt = spielStaerke[j] + spielStaerke[k];
                                        staerkeGeteilt = spielStaerke[j] / spielStaerkeGesamt;

                                        zufall = Math.random();

                                        // zufall wird ausgelesen (s. Dokumentation)
                                        if (zufall < staerkeGeteilt)
                                                ++siege[j];
                                        else
                                                ++siege[k];
                                }
                        }
                }

                // Feststellung der Spieler mit den meisten Siegen der einzelnen Kämpfe
                for (int i = 0; i < spielerAnzahl; ++i)
                {
                        if (siege[i] > siegeVergleich)
```

```
                        {
                                siegeVergleich = siege[i];

                                // Spieler mit den meisten Siegen
                                meisteSiege = i;
                        }
                }

                // siegeGesamt wird für den Spieler mit den meisten Siegen des Turniers erhöht
                ++siegeGesamt[meisteSiege];
        }
```

Q2: Turniere werden durchgeführt.

Im Anschluss daran werden die Siege eines jeden Spielers ausgegeben. Dabei steht der Wert des Spielers mit der höchsten Spielstärke, der vorhin ermittelt wurde, noch einmal hervorgehoben.

Beispiele:

```
Wieviele Spieler nehmen an dem Turnier teil? 8
Ueber wieviele Runden soll das Turnier gespielt werden? 1000
Wie stark ist Spieler 1? 0
Wie stark ist Spieler 2? 10
Wie stark ist Spieler 3? 20
Wie stark ist Spieler 4? 30
Wie stark ist Spieler 5? 40
Wie stark ist Spieler 6? 50
Wie stark ist Spieler 7? 60
Wie stark ist Spieler 8? 100
Staerkster: Siege von Spieler 8: 508
Siege von Spieler 1 : 0
Siege von Spieler 2 : 0
Siege von Spieler 3 : 12
Siege von Spieler 4 : 40
Siege von Spieler 5 : 90
Siege von Spieler 6 : 152
Siege von Spieler 7 : 198
Siege von Spieler 8 : 508
```

Abb. 1: Spielstärken1 bei 1000 Runden

```
Wieviele Spieler nehmen an dem Turnier teil? 8
Ueber wieviele Runden soll das Turnier gespielt werden? 1000
Wie stark ist Spieler 1? 10
Wie stark ist Spieler 2? 10
Wie stark ist Spieler 3? 10
Wie stark ist Spieler 4? 10
Wie stark ist Spieler 5? 80
Wie stark ist Spieler 6? 80
Wie stark ist Spieler 7? 80
Wie stark ist Spieler 8? 100
Staerkster: Siege von Spieler 8: 252
Siege von Spieler 1 : 1
Siege von Spieler 2 : 0
Siege von Spieler 3 : 0
Siege von Spieler 4 : 0
Siege von Spieler 5 : 290
Siege von Spieler 6 : 229
Siege von Spieler 7 : 228
Siege von Spieler 8 : 252
```

Abb. 2: Spielstärken2 bei 1000 Runden

K.O.:

Lösungsidee:

Das Prinzip ist weitestgehend das gleiche wie bei dem System Liga, nur werden die Turniere anders ausgetragen und die Anzahl der Spiele pro Turnier wird am Anfang ermittelt.
Die Anzahl der Spiele pro Turnier wird wie folgt ermittelt: Da sich die Anzahl der Spieler jede Runde im Turnier halbiert, ist die Anzahl der Spiele pro Turnier quadriert gleich der Anzahl der Spieler. Damit kann man auch die Spielanzahl ermitteln.
Zuerst werden alle Spieler mit ihrer Nummer in einem Feld gespeichert, damit jeder ausgeschiedene Spieler entfernt werden kann.
Um den Sieger eines Turniers zu ermitteln, spielen jeweils zwei Spieler gegeneinander. Dabei wird das gleiche Prinzip verwendet wie bei Liga. Die Spieler, die gewinnen, werden im Feld gespeichert, sodass es alle verbleibenden Spieler enthält. Danach wird dieses Prinzip mit den allen verbleibenden Spielern wiederholt.

Umsetzung:

Ich werde hier nur die Dinge erklären, die sich hier vom System Liga unterscheiden. Es gibt einige neue Variablen, alle anderen erfüllen den gleichen Zweck wie im System Liga und werden auch so wie oben beschrieben definiert:

1. mitspieler[] hält alle Spieler fest, welche noch nicht verloren haben, also noch im Spiel sind.
2. mitspielerNummer gibt an, an welcher Stelle mitspieler[] ausgelesen werden soll und ist beim Austragen der Turniere wichtig.
3. spiele gibt an, wieviele Spiele pro Turnier ausgetragen werden müssen, damit nur noch ein Spieler übrig ist, da sich diese Zahl je nach Spieleranzahl ändert.
4. ersterSpieler und zweiterSpieler sind Spielers eines Spiels und dienen nur zur Übersicht, damit nicht zu viele Variablen ineinander verschachtelt sind.

Nach der Deklaration und ggf. Definition der Variablen werden zuerst die Mitspieler an ihrer Stelle i auf den Wert i gesetzt (Abb. 4).
Dann wird die Anzahl der benötigten Spiele ermittelt. Dazu wird eine for-Schleife geöffnet, die bis zur Hälfte der Anzahl der Spieler geht, denn mehr Runden kann es bei einem Turnier nicht geben. Wenn 2 hoch i gleich der Spieleranzahl ist, wird spiele auf diesen Wert gesetzt (s. Lösungsidee) (Q3).

```
	//Ermittlung der Spielanzahl
	for (int i = 0; i < spielerAnzahl / 2; ++i)
	{
		if (Math.pow(2, i) == spielerAnzahl)
		{
			spiele = i;
			break;
		}
	}
```

Q3: Ermittlung der Spielanzahl

Danach werden – wie beim System Liga – die Werte der Spieler eingelesen und die größte Spielerstärke wird ermittelt. Um die Turniere auszutragen, wird bei jeder Wiederholung mitspieler[] wieder zurückgesetzt, da sich dann alle Spieler wieder im Spiel befinden. Dies ist allerdings nur bei Wiederholungen wichtig, nicht, wenn nur ein Turnier ausgetragen wird.
Nach dem wird dann das Turnier ausgetragen. Zuerst wird eine for-Schleife geöffnet, welche die nachfolgenden Schritte so oft wiederholt, wie in der Variable spiele festgehalten ist.
Es wird eine weitere for-Schleife geöffnet, die für die Anzahl aller verbleibenden Spieler wiederholt wird. Diese Anzahl wird ermittelt, indem man die Spieleranzahl durch zwei hoch i teilt, da sie sich in jedem Spiel halbiert. Am Anfang ist sie also bei 16 Spielern 16, dann acht, vier, zwei bis eins.
Danach werden noch ersterSpieler und zweiterSpieler festgelegt. Diese werden auf den Wert von mitspieler[] an der Stelle j gesetzt, sodass sie als Wert die Nummer eines Spielers haben, also z.B. wenn Spieler Eins spielt, beträgt der Wert für ersterSpieler 0, wenn aber Spieler Fünf spielt, ist der Wert 4.
Im Anschluss werden spielStaerkeGesamt, staerkeGeteilt und zufall genauso bestimmt wie im System Liga.
Danach wird ermittelt, welcher Spieler gewonnen hat. Die Variable mitspieler[] wird an der Stelle mitspielerNummer auf den Gewinner gesetzt. Dann wird mitspielerNummer erhöht, sodass der Wert bei einem Spiel nicht überschrieben wird. Nachdem zwei Spieler gegeneinander gespielt haben, wird j um zwei erhöht, sodass jeder Spieler pro Spiel nur einmal antritt.
Hat dann jeder Spieler gespielt, hat sich die Anzahl der verbleibenden Spieler halbiert.
Dann wird mitspielerNummer auf 0 gesetzt, um das nächste Spiel austragen zu können, in dem aus den verbleibenden Spielern wieder die Hälfte entfernt wird. Danach wird dieses Prinzip für die Anzahl der Spiele wiederholt, jedes Mal allerdings nur halb so oft wie im Spiel davor, sodass am Ende nur noch ein Spieler übrig ist. Dieser ist in mitspieler[0] gespeichert und die Variable siege wird für diesen Spieler erhöht.
Sollte es Wiederholungen geben, wird ein weiteres Turnier ausgetragen und alle damit einhergehenden Schritte werden durchgeführt. Der Teil des Codes zum Austragen der Turniere ist in Q4 zu sehen.

```
	// Austragen der Turniere
	for (int a = 0; a < wiederholungen; ++a)
	{

		for (int i = 0; i < spielerAnzahl; ++i)
		{
			// Zuruecksetzen der Mitspieler
			mitspieler[i] = i;
		}

		for (int i = 0; i < spiele; ++i)
		{
			for (int j = 0; j < (spielerAnzahl / (Math.pow(2, i))); j += 2)
			{
				// fuer alle (verbleibenden) Mitspieler wird das Turnier durchgeführt
				ersterSpieler = mitspieler[j];
				zweiterSpieler = mitspieler[(j + 1)];
				spielStaerkeGesamt = spielStaerke[ersterSpieler] + spielStaerke[zweiterSpieler];
				staerkeGeteilt = spielStaerke[ersterSpieler] / spielStaerkeGesamt;
				zufall = Math.random();
				if (zufall < staerkeGeteilt)
				{
					// zufall wird ausgelesen (s. Dokumentation)
					mitspieler[mitspielerNummer] = ersterSpieler;
				}
				else
				{
					mitspieler[mitspielerNummer] = zweiterSpieler;
				}
				++mitspielerNummer;
			}
			mitspielerNummer = 0;
		}
		// Erhoehung der Siege des Spielers
		++siege[mitspieler[0]];
	}
```

Q4: Austragen der Turniere

Nun werden nur noch die Werte der Siege wie im System Liga ausgegeben.

Beispiele:

```
Wieviele Spieler nehmen an dem Turnier teil? 8
Ueber wieviele Runden soll das Turnier gespielt werden? 1000
Wie stark ist Spieler 1? 0
Wie stark ist Spieler 2? 10
Wie stark ist Spieler 3? 20
Wie stark ist Spieler 4? 30
Wie stark ist Spieler 5? 40
Wie stark ist Spieler 6? 50
Wie stark ist Spieler 7? 60
Wie stark ist Spieler 8? 100
Staerkster: Siege von Spieler 8: 337
Siege von 1: 0
Siege von 2: 32
Siege von 3: 53
Siege von 4: 138
Siege von 5: 115
Siege von 6: 168
Siege von 7: 157
Siege von 8: 337
```

Abb. 3: Spielstärken1 bei 1000 Runden

```
Wieviele Spieler nehmen an dem Turnier teil? 8
Ueber wieviele Runden soll das Turnier gespielt werden? 1000
Wie stark ist Spieler 1? 10
Wie stark ist Spieler 2? 10
Wie stark ist Spieler 3? 10
Wie stark ist Spieler 4? 10
Wie stark ist Spieler 5? 80
Wie stark ist Spieler 6? 80
Wie stark ist Spieler 7? 80
Wie stark ist Spieler 8? 100
Staerkster: Siege von Spieler 8: 278
Siege von 1: 21
Siege von 2: 28
Siege von 3: 22
Siege von 4: 28
Siege von 5: 202
Siege von 6: 219
Siege von 7: 202
Siege von 8: 278
```

Abb. 4: Spielstärken2 bei 1000 Runden

K.O.x5:

Lösungsidee:

Das Prinzip hier ähnelt dem von K.O. sehr, nur, dass zwei Spieler nicht einmal gegeneinander spielen, sondern fünf mal. Der Spieler, der die meisten Spiele gewonnen hat, kommt weiter, der andere scheidet aus.

Umsetzung:

Um das Prinzip mit fünf Partien pro Runde zu realisieren, habe ich die Variable siege[] aus dem System K.O. umfunktioniert. Sie hält nun für jeden Spieler fest, wieviele Male er in einer Partie gewonnen hat. Dafür hält spieleGesamt[] fest, wie oft ein Spieler ein ganzes Turnier gewonnen hat.
Bis zum Austragen der Turniere ist der Quellcode – bis auf die Deklaration der Variable siegeGesamt[] - der gleiche wie im System K.O.
Werden nun die Turniere ausgetragen, wird zuerst neben mitspieler[] auch siege[] auf den Standardwert (bei siege[] ist das 0) gesetzt, da diese Werte nur für ein Spiel gelten.
Dann werden die Variablen ersterSpieler, zweiterSpieler, spielStaerkeGesamt und staerkeGeteilt wie im System K.O. definiert.
Im Anschluss wird eine for-Schleife geöffnet, welche sich fünf mal wiederholt. Dabei wird mit der gleichen Methode wie in den beiden vorhergehenden Systemen der Sieger ermittelt. Diesmal wird aber nur die Variable siege[] für den betroffenen Spieler erhöht.
Nun wird die Anzahl der Siege verglichen. Mit dem Sieger und Verlierer des Vergleiches wird genauso verfahren wie im System K.O. Der Verlierer scheidet also aus, der Gewinner bleibt im Spiel.
Nun werden noch mitspielerNummer[] und siegeGesamt[] des Spielers, der gewonnen hat, erhöht.
Dieser Teil des Codes ist in Q5 zu sehen.

```
        // fuer alle (verbleibenden) Mitspieler wird das Turnier durchgeführt
        for (int j = 0; j < (spielerAnzahl / (Math.pow(2, i))); j += 2)
        {
                ersterSpieler = mitspieler[j];
                zweiterSpieler = mitspieler[(j + 1)];
                spielStaerkeGesamt = spielStaerke[ersterSpieler] + spielStaerke[zweiterSpieler];
                staerkeGeteilt = spielStaerke[ersterSpieler] / spielStaerkeGesamt;

                // Durchfuehren der Spiele im Turnier
                for (int k = 0; k < 5; ++k)
                {
                        zufall = Math.random();
                        // zufall wird ausgelesen (s. Dokumentation)
                        if (zufall < staerkeGeteilt)
                        {
                                ++siege[ersterSpieler];
                        }
                        else
                        {
                                ++siege[zweiterSpieler];
                        }

                        if (siege[ersterSpieler] > siege[zweiterSpieler])
                        {
                                mitspieler[mitspielerNummer] = ersterSpieler;
                        }
                        else
                        {
                                mitspieler[mitspielerNummer] = zweiterSpieler;
                        }
                        ++mitspielerNummer;
                }

                // Erhoehung der Siege des Spielers
                ++siegeGesamt[mitspieler[0]];
        }
```

Q5: Austragen der Turniere

Nun werden nur noch die Werte wie in den beiden anderen Systemen ausgegeben.

Beispiele:

```
Wieviele Spieler nehmen an dem Turnier teil? 8
Ueber wieviele Runden soll das Turnier gespielt werden? 1000
Wie stark ist Spieler 1? 0
Wie stark ist Spieler 2? 10
Wie stark ist Spieler 3? 20
Wie stark ist Spieler 4? 30
Wie stark ist Spieler 5? 40
Wie stark ist Spieler 6? 50
Wie stark ist Spieler 7? 60
Wie stark ist Spieler 8? 100
Staerkster: Siege von Spieler 8: 560
Siege von 1: 0
Siege von 2: 9
Siege von 3: 12
Siege von 4: 120
Siege von 5: 44
Siege von 6: 99
Siege von 7: 156
Siege von 8: 560
```

Abb. 5: Spielstärken1 bei 1000 Runden

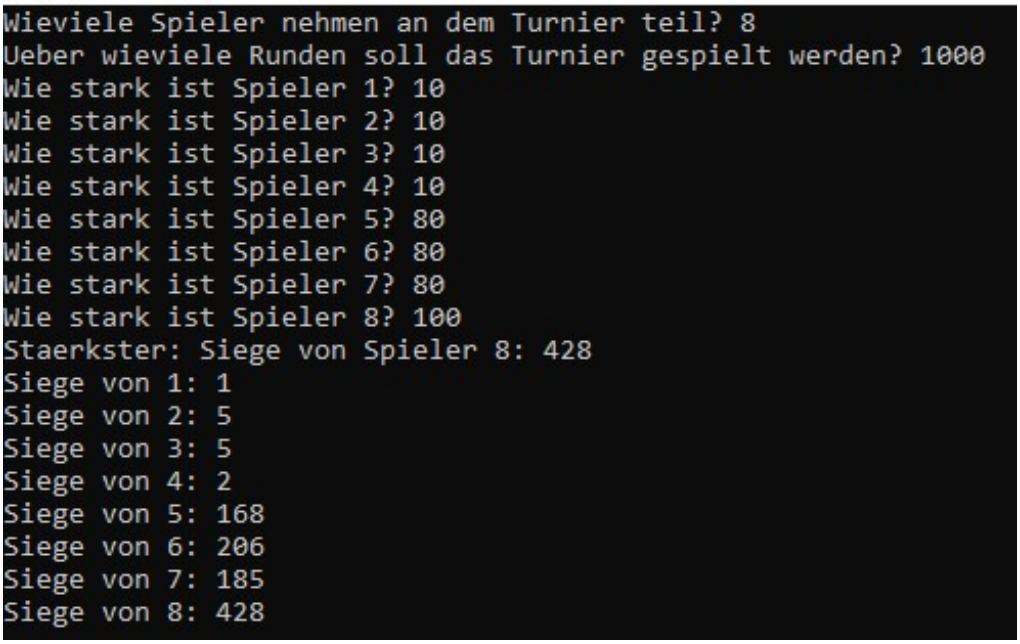

Abb. 6: Spielstärken2 bei 1000 Runden

Fazit:

Ich empfehle Tobi das System Liga, da nach meinen Ergebnissen in diesem System die Spielstärke am stärksten mit den Siegen korreliert. Bei den anderen Systemen können Spieler durch ihre Position und damit ihre Gegner in Partien ungerecht bevorteilt werden.