**Chuck a Luck**

## Zielstellung

Diese Aufgabe soll weniger dem Erlernen neuer Inhalte dienen, sondern ist als Anwendungs- bzw. Projektaufgabe gedacht.

## Material

Aufgabenstellung, Lösungsvorschlag und BlueJ-Projekte (*chuckaluck_6a – 6d*) können aus dem Internet heruntergeladen werden:
https://www.isb.bayern.de/gymnasium/uebersicht/kompetenzorientierte-aufgaben-informatik/

## Aufgabe

Chuck a Luck, was zu Deutsch etwa „Glückswurf" bedeutet, ist ein einfaches Würfelglücksspiel mit drei Würfeln, das in vielen Kasinos der Welt gespielt wird. Dabei setzt jeder der Spieler zunächst einen selbst gewählten Einsatz auf eine der Zahlen eins bis sechs, anschließend wirft der Spielleiter die drei Würfel. Zeigt nun mindestens ein Würfel die gesetzte Zahl, bekommt der Spieler seinen Einsatz zurück und zusätzlich einen Einsatz für jeden Würfel, der seine Zahl zeigt. D. h. er gewinnt seinen Einsatz, wenn ein Würfel die gesetzte Zahl zeigt, er gewinnt den doppelten Einsatz, wenn zwei Würfel die gesetzte Zahl zeigen, und er gewinnt den dreifachen Einsatz, wenn alle drei Würfel die gesetzte Zahl zeigen. Zeigt kein Würfel die gesetzte Zahl, so ist der Einsatz verloren.

a) Entwickeln Sie in Kleingruppen eine Softwaresimulation von Chuck a Luck für einen oder mehrere (beliebig viele) Spieler. Entwerfen Sie zunächst ein Klassendiagramm mit allen Attributen und Methoden und implementieren Sie dann die einzelnen Klassen arbeitsteilig. Vereinbaren Sie vorab verbindliche Termine für die Fertigstellung und verfassen Sie am Ende eine Dokumentation Ihrer Software.

b) Nun soll simuliert werden, wie hoch der durchschnittliche Gewinn des Kasinos bei diesem Spiel ist. Erweitern Sie dazu Ihre Simulation so, dass ein Spieler automatisch 100.000-mal spielt und dabei jedes Mal einen Euro setzt.

   Begründen Sie, wie sich das Ergebnis ändert, wenn der Spieler bei jedem Spiel auf dieselbe Zahl setzen würde.
   Das Simulationsergebnis soll nun rechnerisch überprüft werden. Versuchen Sie, eine Formel herzuleiten, indem Sie zunächst berechnen, wie wahrscheinlich es ist, drei Euro, zwei Euro und einen Euro zu gewinnen bzw. einen Euro zu verlieren. Überlegen Sie dann, wie hoch Ihr Gewinn pro Spiel im Schnitt sein wird.

c) Durch Hinzufügen einer weiteren Wettmöglichkeit, dem sog. „Field", entsteht aus Chuck a Luck das Kasinospiel Mini Dice. Bei „Field" wettet der Spieler darauf, dass die Augensumme der drei Würfel einen der Werte 5, 6, 7, 8, 13, 14, 15 oder 16 annimmt. Im Gewinnfall erhält er seinen Einsatz zurück und einen weiteren Einsatz zusätzlich, d. h., er gewinnt einen Einsatz. Fügen Sie Ihrer Simulation von Teilaufgabe a die Wettmöglichkeit „Field" hinzu.

a) Auch bei „Field" gewinnt im Schnitt immer das Kasino. Simulieren Sie wieder wie in Teilaufgabe b den durchschnittlichen Gewinn eines Spielers, der 100.000-mal mit einem Einsatz von einem Euro spielt.

   Versuchen Sie wieder, Ihr Simulationsergebnis rechnerisch zu überprüfen. Überlegen Sie sich dazu, wie viele Möglichkeiten es gibt, mit drei Würfeln die Augensummen 5, 6, 7 etc. zu erzielen, und wie viele Möglichkeiten es insgesamt gibt. Falls Ihnen das Zählen der Möglichkeiten zu mühsam ist, können Sie es auch dem Computer überlassen, indem Sie ein kleines Programm dafür entwickeln.

**Hinweise zur Aufgabe**

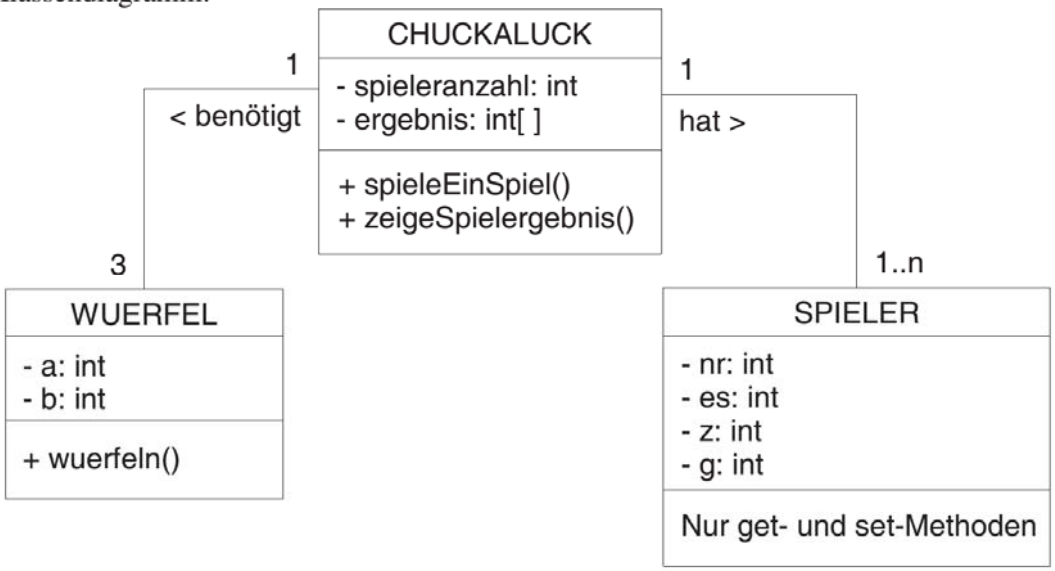
Die Lösung der Teilaufgabe a kann auch umfangreicher ausfallen. Denkbar wäre etwa, dass mehrere Spiele von denselben Spielern gespielt werden und die Attribute es (Einsatz), z (gesetzte Zahl) und g (Gewinn) des Spielers und das Ergebnis eines Spieles jeweils abhängig von einer Spielnummer in einer anderen Klasse implementiert werden.

Die rechnerische Überprüfung der Simulationsergebnisse in den Teilaufgaben b und d ist jeweils eine fächerübergreifende Aufgabenstellung mit mathematischen Inhalten und kann auch weggelassen werden. Alternativ wäre es möglich, die Erwartungswerte anzugeben und den Schülerinnen und Schülern die Aufgabe zu geben, sie herzuleiten. Ebenso denkbar wäre, dass die Schülerinnen und Schüler die Erwartungswerte mithilfe einer Internetrecherche selber finden und anschließend verifizieren.

**Lösungshinweise**

a)

Klassendiagramm:




**Implementierung der Klasse CHUCKALUCK**

```
public class Chuckaluck {
    private Wuerfel[] wuerfel;
    private int spielerzahl;
    private Spieler[] spieler;
    private int[] ergebnis;

    public Chuckaluck(int anzahlSpieler) {
        wuerfel = new Wuerfel[3];
        ergebnis = new int[3];
        spielerzahl = anzahlSpieler;
        spieler = new Spieler[spielerzahl];
        for (int i=0; i<3; i++) {
            wuerfel[i] = new Wuerfel();
        }
        for (int i=0; i<spielerzahl; i++) {
            spieler[i] = new Spieler(i);
        }
    }

    public void spieleEinSpiel() {
        int[] treffer = new int[spielerzahl];
        for (int i=0; i<3; i++) {

            ergebnis[i] = wuerfel[i].wuerfeln();
        }
        for (int i=0; i<spielerzahl; i++) { treffer[i] = 0;
            for (int k=0; k<3; k++) {
                if (ergebnis[k]==spieler[i].gibZahl()) {
                    treffer[i] = treffer[i] + 1;
                }
            }
            if (treffer[i]>0) {
                spieler[i].setzeGewinn(treffer[i]*spieler[i].gibEinsatz());
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }
        }
    }

    public void zeigeSpielergebnis() {
        System.out.println("Spielergebnis:");
        for (int i=0; i<3; i++) {
            System.out.println("Würfel " + i + ": " + ergebnis[i]);
        }
        for (int i=0; i<spielerzahl; i++) {
```

```
            System.out.println("Spieler " + i + ": gesetze Zahl " +
              spieler[i].gibZahl() + " / Einsatz " + spieler[i].gibEinsatz() +
              " / Gewinn " + spieler[i].gibGewinn());
        }
    }
}
```

b) Es kommt eine weitere Klasse SIMULATION hinzu:

```
public class Simulation {
    private Chuckaluck chuckaluck;

    public Simulation() {
        chuckaluck = new Chuckaluck();
    }

    public void simuliere(int anzahlSpiele) {
        int gesamtgewinn = 0;
        for (int i=0; i<anzahlSpiele; i++) {
            chuckaluck.spieleEinSpiel();
            gesamtgewinn = gesamtgewinn +
            chuckaluck.gibErstenSpieler().gibGewinn();
        }
        System.out.println("Der Spieler hat " + gesamtgewinn + " gewonnen.");
    }
}
```

Das Ergebnis würde sich nicht ändern, wenn man bei jedem Spiel auf dieselbe Zahl setzen würde, denn die Würfel haben kein „Gedächtnis“.

Rechnerische Überprüfung: Erwarteter Gewinn pro Spiel bei einem Einsatz von einem Euro:

$$\left(\frac{1}{6}\right)^3 \cdot 3€ + 3 \cdot \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6} \cdot 2€ + 3 \cdot \frac{1}{6} \cdot \left(\frac{5}{6}\right)^2 \cdot 1€ + \left(\frac{5}{6}\right)^3 \cdot (-1€) =$$

$$\frac{1}{216} \cdot 3€ + \frac{15}{216} \cdot 2€ + \frac{75}{216} \cdot 1€ + \frac{125}{216} \cdot (-1€) = -\frac{17}{216}\ € \approx -0{,}07870€$$

Bei 100.000 Spielen würde man also im Schnitt 7870€ verlieren.

c) Angepasste Methoden der Klasse CHUCKALUCK:

```
public void spieleEinSpiel() {
    augensumme = 0;
    int[] treffer = new int[spielerzahl];
    for (int i=0; i<3; i++) {
        ergebnis[i] = wuerfel[i].wuerfeln();
        augensumme = augensumme + ergebnis[i];
    }
    for (int i=0; i<spielerzahl; i++) {
        if (!spieler[i].gibField()) {
            treffer[i] = 0;
            for (int k=0; k<3; k++) {
                if (ergebnis[k]==spieler[i].gibZahl()) {
                    treffer[i] = treffer[i] + 1;
                }
            }
            if (treffer[i] > 0) {
                spieler[i].setzeGewinn (treffer[i]*spieler[i].gibEinsatz());
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }

        } else {
            if ((augensumme > 4 && augensumme < 9) ||
                (augensumme > 12 && augensumme < 17)) {
                spieler[i].setzeGewinn(spieler[i].gibEinsatz());
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }
```

```
        }
    }
}

public void zeigeSpielergebnis() {
    System.out.println("Spielergebnis:");
    for (int i=0; i<3; i++) {
        System.out.println("Würfel " + i + ": " + ergebnis[i]);
    }
    System.out.println("Augensumme: " + augensumme);
    for (int i=0; i<spielerzahl; i++) {
        if (!spieler[i].gibField()) {
            System.out.println("Spieler " + i + ": gesetze Zahl " +
            spieler[i].gibZahl() + " / Einsatz " + spieler[i].gibEinsatz()+
            " / Gewinn " + spieler[i].gibGewinn());
        } else {
            System.out.println("Spieler " + i + ": auf Field gesetzt"+
              " / Einsatz "+ spieler[i].gibEinsatz()+
              " / Gewinn " + spieler[i].gibGewinn());
        }
    }
}
```

d) Die Klasse SIMULATION ändert sich gegenüber Teilaufgabe b nicht, die Klassen CHUCKALUCK und SPIELER erhalten gegenüber Teilaufgabe c zusätzliche Konstruktoren, ähnlich wie in Teilaufgabe b.

Rechnerische Überprüfung:
Anzahl der Möglichkeiten mit drei (unterscheidbaren) Würfeln eine bestimme Augensumme zu erzielen:

| Augensumme | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Möglichkeiten | 1 | 3 | 6 | 10 | 15 | 21 | 25 | 27 | 27 | 25 | 21 | 15 | 10 | 6 | 3 | 1 |

Insgesamt gibt es $6^3 = 216$ Möglichkeiten, d. h. die Wahrscheinlichkeit zu gewinnen beträgt:

$$\frac{6}{216}+\frac{10}{216}+\frac{15}{216}+\frac{21}{216}+\frac{21}{216}+\frac{15}{216}+\frac{10}{216}+\frac{6}{216}=\frac{104}{216}\approx 0{,}4815$$

Damit beträgt die Wahrscheinlichkeit zu verlieren:

$$1-\frac{104}{216}=\frac{112}{216}\approx 0{,}5185$$

Man erwartet also folgenden Gewinn pro Spiel bei einem Einsatz von einem Euro:

$$\frac{104}{216}\cdot 1€+\frac{112}{216}\cdot(-1€)=-\frac{8}{216}€\approx -0{,}03704€$$

Bei 100.000 Spielen würde man also im Schnitt 3704€ verlieren.

Die oben aufgeführte Wahrscheinlichkeitsverteilung lässt sich schnell mithilfe eines kleinen Java-Programms berechnen:

```
public class Wvert {
    private int[] augensumme;

    public Wvert() {
        augensumme = new int[19];
        for (int i=0; i<19; i++) {
            augensumme[i] = 0;
        }
    }

    public void verteilungBerechnen() {
        int summe;
        for (int i=1; i<7; i++) {
            for (int j=1; j<7; j++) {
                for (int k=1; k<7; k++) {
                    summe = i + j + k;
                    augensumme[summe] = augensumme[summe] + 1;
```

```
                }
            }
        }
    }

    public void verteilungAusgeben() {
        for (int i=3; i<19; i++) {
            System.out.println("Anzahl der Möglichkeiten für Augensumme " +
            i + ": " + augensumme[i]);
        }
    }
}
```

*Quelle: Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.). Kompetenzorientierte Aufgaben für das Fach Informatik am Gymnasium. München: Brigg Pädagogik Verlag, 2013, S.143-149 (bearbeitet)*

**Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen**

| Aufg. | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| a | X | | | X | X | X | X | | X | | | X | X |
| b | X | X | | | | | X | | | | | X | X |
| c | X | | | X | | | X | | X | | | X | |
| d | X | | | | | | X | | | | | X | X |