Kontrollstrukturen

Verzweigungen und Schleifen

Inhaltsverzeichnis

Rock Paper Scissors

```
import java.util.Scanner;

public class RockPaperScissors {

	public static void main(String[] args) {

		Scanner scanner = new Scanner(System.in);

		// Punktestand. Es wird gespielt bis einer 2 Punkte hat
		int scoreUser = 0;
		int scorePC = 0;

		while (scoreUser < 2 && scorePC < 2) {

			// Eingabe Benutzer und Zufallszahl Computer
			System.out.println("1 = Rock");
			System.out.println("2 = Paper");
			System.out.println("3 = Scissors");
			System.out.print("Welches Symbol wollen Sie nutzen? ");
			int user = scanner.nextInt();
			int pc = (int)(Math.random()*3+1);

			// Ausgabe Symbol Computer
			if (pc == 1)
				System.out.println("Der Computer nutzt Rock.");
			if (pc == 2)
				System.out.println("Der Computer nutzt Paper.");
			if (pc == 3)
				System.out.println("Der Computer nutzt Scissors.");

			// Ausgabe Symbol Benutzer
			if (user == 1)
				System.out.println("Sie nutzen Rock.");
			if (user == 2)
				System.out.println("Sie nutzen Paper.");
			if (user == 3)
				System.out.println("Sie nutzen Scissors.");
```

```
                        // Ausgabe Spielausgang, Aktualisierung Spielstand
                        if ((3+pc-user)%3==0) {
                                System.out.println("Unentschieden.");
                        }
                        if ((3+pc-user)%3==1) {
                                System.out.println("Der Computer gewinnt.");
                                ++scorePC;
                        }
                        if ((3+pc-user)%3==2) {
                                System.out.println("Sie gewinnen.");
                                ++scoreUser;
                        }
                        System.out.println("Spielstand "+scoreUser+":"+scorePC);

                }

                scanner.close();

        }

}
```

Bisektionsverfahren

```
public class Bisection {

        public static void main(String[] args) {

                double a = 0;
                double b = 1;

                double m;
                double f;

                do {

                        m = (a+b)/2;
                        f = m*m*m + 3*m*m - 1;

                        if (f < 0)
                                a = m;
                        else
                                b = m;

                } while (Math.abs(f) > 0.01);

                System.out.println("Nullstelle (x-Wert): " + m);

        }

}
```

Die nachfolgend präsentierte Lösung für das Bisektionsverfahren verwendet bereits Methoden und ist flexibel gehalten: Sowohl die Funktion als auch das Intervall können mit wenigen Änderungen im Quelltext ausgetauscht werden.

```
public class Bisection {

	static final double EPS = 0.01;

	// Funktion, von der die Nullstelle berechnet werden soll
	public static double f(double x) {
		return x*x*x + 3*x*x - 1;
	}

	public static double bisection(double a, double b) {

		// Intervallmitte
		double m = (a+b)/2;

		// Wo ist der Funktionswert positiv
		boolean apos = f(a) > 0;
		boolean bpos = f(b) > 0;
		boolean mpos = f(m) > 0;

		// Ist der Funktionswert nahe genug bei 0, wird abgebrochen
		while (Math.abs(f(m)) > EPS) {
			if (apos == mpos)
				a = m;
			if (bpos == mpos)
				b = m;
			m = (b+a)/2;
			mpos = f(m) > 0;
		}

		return m;
	}

	public static void main(String[] args) {
		double ns = bisection(0,1);
		System.out.println("Nullstelle: (" + ns + "," + f(ns) + ")");
	}

}
```

Betrunkener

```
public class DrunkenStudent {
	public static void main(String[] args) {
		int room = 7;
		int counter = 0;

		while (room != 1) {

			System.out.print("Zimmer "+room);
			if (Math.random() < 0.5) {
				++room;
				System.out.print(" -> Kopf -> ");
			}
			else {
				--room;
				System.out.print(" -> Zahl -> ");
			}

			// Alle 4 Schritte einen Zeilenumbruch auf der Konsole erzwingen
			++counter;
			if (counter%4 == 0)
				System.out.println();

		}

		System.out.println("Endlich zurück.");
		System.out.println("Nach "+counter+" Versuchen");
	}
}
```

Halbieren

Gleitkommazahlen können bei einer Halbierung entweder den Exponenten um 1 reduzieren (das ist 2^(Anzahl Exponenten-Bit minus 1) mal möglich) oder die Mantisse halbieren, also das mit 1 besetzte bit um eine Position nach rechts verschieben (das ist Anzahl Mantisse-Bit minus 1 mal möglich.

Insgesamt ergibt sich also, dass 2^(Exp-1) + Mant-1 mal halbiert werden kann. Für float sind das 2^7+22 = 150 mal, für double sind es 2^10+51 = 1075 mal.

```
public class HalveDatentyp {
	public static void main(String[] args) {
		datentyp x = 1;
		int counter = 0;
		while (x > 0) {
			x /= 2;
			++counter;
		}
		System.out.println(counter);
	}
}
```