Programmieren 1
Lernteam Semesterwoche 6

Aufgabe 1: Arrays
Lösen Sie die Aufgaben 4.44 bis 4.49 im Lehrbuch.

Aufgabe 2: Arrays, Methoden und Debugger
Übernehmen Sie folgendes Gerüst für eine Klasse Aufgabe2:

```
public class Aufgabe2
{
	private int[] a;
	public Aufgabe2(int length)
	{
		a = new int[length];
	}
	//...
}
```

Implementieren Sie folgende Methoden:
a) public void setFix(int value): Initialisiert alle Felder von a mit dem Wert von value.

```
public void setFix(int value)
{
	for(int i = 0; i<a.length; i++)
	{
		a[i] = value;
	}
}
```

b) public void setUp(): Initialisiert a aufsteigend mit Werten 1, 2, 3, …

```
public void setUp()
{
	for(int i = 0; i<a.length; i++)
	{
		a[i] = i+1;
	}
}
```

c) public void setRandom(int min, int max): Initialisiert a zufällig mit Werten aus dem Bereich [min, max] (vgl. mit dem Beispiel aus der Vorlesung).

```
public void setRandom(int min, int max)
{
	for(int i = 0; i<a.length; i++)
	{
		// Generiert eine zufällige Zahl zwischen 0.0 und <1.0
		double r = Math.random();

		// Generiert eine zufällige Zahl zwischen min und max
		int n = (int) (r * (max - min +1) + min);

		a[i] = n;
	}

}
```

d) public void add(int value): Addiert zu allen Feldern von a den Wert value.

```
public void add(int value)
{
	for(int i = 0; i<a.length; i++)
	{
		a[i] = a[i] + value;
	}
}
```

e) public boolean add(int[] b): Addiert b elementweise zu a (falls überhaupt möglich!). Bei erfolgreicher Addition ist der Rückgabewert true.

```
public boolean add(int[] b)
{
	boolean returnvalue = false;

	for(int i = 0; i<b.length; i++)
	{
		if(i<a.length)
		{
			a[i] = a[i] + b[i];
			returnvalue = true;
		}
		else
		{
			returnvalue = false;
			break;
		}
	}
	return returnvalue;
}
```

Überprüfen Sie das Funktionieren Ihrer Methoden mit Hilfe des Object-Inspectors und Debuggers von BlueJ. Hinweis: Ein Array lässt sich in BlueJ via Eingabe-Fenster direkt wie folgt als Parameter einer Methode übergeben, z.B.: {1, 4, 7, -10}
➔ Methode getestet mit a[2] und b[4]. Nur die ersten zwei Elemente werden addiert. Nachher wird abgebrochen, da Array a weniger Elemente hat als Array b.

Aufgabe 3: Gleichwertigkeit der Schleifen-Anweisungen

Lösen Sie die Aufgabe 4.50 im Lehrbuch (while & for).

Aufgabe 4: Klasse GuestBook

Implementieren Sie eine einfache Klasse GuestBook. Ein Gästebuch kann beliebig viele Einträge (vgl. Zeichenketten in einer ArrayList) aufnehmen. Die Anzahl Einträge können abgefragt und das Gästebuch kann auf die Konsole zeilenweise ausgegeben werden. Entsprechend stehen folgende Methoden im Zentrum:
a) public void insertEntry(String note)

b) public int getNoOfEntries()
c) public void printEntries()

```
public class Guestbook
{
    private String entries[];

    public Guestbook()
    {
        entries = new String[0];
    }

    public void insertEntry(String note)
    {
        String extended[] = new String[entries.length+1];
```

```
        System.arraycopy(entries, 0, extended, 0, entries.length);

        extended[extended.length-1] = note;

        entries = new String[extended.length];
        System.arraycopy(extended, 0, entries, 0, extended.length);

    }

    public int getNoOfEntries()
    {
        return entries.length;
    }

    public void printEntries()
    {
        for (String str : entries)
        {
            System.out.println(str);
        }
    }
}
```

Optionale Zusatzaufgabe 5: Klasse Ticket

Implementieren Sie eine Klasse Ticket. Ein Ticket besitzt einen Preis (price; nur Anzahl Franken), eine Aufschrift (text; Text-Zeile) und eine bestimmte Breite (width; Anzahl Zeichen) und Höhe (height; Anzahl Zeichen). Die konkreten Werte lassen sich über entsprechende Zugriffsmethoden abfragen. Für Preis und Aufschrift gibt es zudem set-Methoden, die nur sinnvolle Eingaben zulassen. Ein Ticket lässt sich mit einer print-Methode wie folgt auf die Konsole ausgeben:

```
****************************************
* *
* *
* Hochschule Luzern - Special Event *
* Fr. 18.- *
* *
* *
****************************************
```

Wahrscheinlich macht es Sinn, zusätzliche Hilfsmethoden (vgl. private) zu implementieren.

```
public class Ticket
{

    private int price = 0;
    private String sprice = "";
    private String text = "";
    private int width = 0;
    private int height = 0;
    private int text_spaces = 0;
    private int price_spaces = 0;

    public Ticket(String text, int price)
    {
        setText(text);
        setPrice(price);
        setDefaultSize();
    }

    public int getPrice()
    {
        return price;
    }

    public String getText()
    {
        return text;
    }

    public int getWidth()
    {
        return width;
    }

    public int getHeight()
    {
        return height;
    }

    public void setText(String text)
    {
        this.text = text;
    }
```

```
public void setPrice(int price)
    {
        this.price = price;
        this.sprice = "Fr. "+price+".-";
    }

    public void setWidth(int width)
    {
        this.width = width;

        if(width>=this.text.length() && width>=this.sprice.length())
        {
            this.text_spaces = width - this.text.length() - 4;
            this.price_spaces = width - this.sprice.length() - 4;
        }
        else if(this.text.length()>=this.sprice.length())
        {
            this.width = this.text.length()+4;
            this.text_spaces = this.sprice.length()-this.text.length();
            this.price_spaces = this.text.length()-this.sprice.length();
        }
        else
        {
            this.width = this.sprice.length()+4;
            this.price_spaces = this.text.length()-this.sprice.length();
            this.text_spaces = this.sprice.length()-this.text.length();
        }
    }

    public void setHeight(int height)
    {
        if(height>=4)
        {
            this.height = height;
        }
        else
        {
            this.height = 4;
        }
    }

    private void setDefaultSize()
    {
        // define default width
        if(this.text.length()>=this.sprice.length())
        {
            this.width = this.text.length()+4;
            this.text_spaces = 0;
            this.price_spaces = this.text.length()-this.sprice.length();
        }
        else
        {
            this.width = this.sprice.length()+4;
            this.price_spaces = 0;
            this.text_spaces = this.sprice.length()-this.text.length();
        }

        // define default height
        this.height = 8;
    }
```

```
public void print()
{
        // First Line
        String line = "";
        for(int x=0; x<width; x++)
        {
            line += "*";
        }
        System.out.println(line);

        // Line between
        for(int y=0; y<(height-4)/2; y++)
        {
            line = "";
            for(int x=0; x<width-6; x++)
            {
                line += " ";
            }
            System.out.println("* *"+line+"* *");
        }

        // Text and Price
        line = "";

            for(int y=0; y<text_spaces; y++)
            {
                line += " ";
            }
            System.out.println("* "+text+line+" *");

        line = "";
            for(int y=0; y<price_spaces; y++)
            {
                line += " ";
            }
            System.out.println("* "+sprice+line+" *");

        // Line between
        for(int y=0; y<(height-4)/2; y++)
        {
            line = "";
            for(int x=0; x<width-6; x++)
            {
                line += " ";
            }
            System.out.println("* *"+line+"* *");
        }

        // Last Line
        line = "";
        for(int x=0; x<width; x++)
        {
            line += "*";
        }
        System.out.println(line);

    }

}
```

Optionale Zusatzaufgabe 6: Klasse PlayDice

Implementieren Sie in der Klasse PlayDice eine Methode public int dice(int noOfSix). Die Methode simuliert das Würfeln mit einem klassischen Würfel (vgl. Augenzahlen 1 bis 6). Genauer würfelt die Methode so lange mit dem Würfel, bis unmittelbar nacheinander noOfSix Mal eine Sechs gefallen ist. Der Rückgabewert entspricht dann der Anzahl erforderlicher Würfe. Experimentieren Sie etwas mit der Methode. Strapazieren Sie aber nicht den Zufall bzw. Ihren Computer! (Inspizieren Sie mal ein PlayDice-Objekt. Wie sieht es mit dem Zustand dieses Objektes aus?)

```
public class PlayDice
{
    public PlayDice()
    {

    }

    public int dice(int noOfSix)
    {
        int[] a = new int[noOfSix];
        int count = 0;
        int n=0;

        for(int i : a)
        {
            do
            {
                n = (int) (Math.random() * 7);
                count++;
            }
            while(n!=6);
        }
        return count;
    }
}
```

Optionale Zusatzaufgabe 7: Klasse Test

Die Klasse Test ermöglicht es, für einen bestimmten Test die Ergebnisse der einzelnen Studierenden zu erfassen. Dabei wird mit Punkten gerechnet (0 …). Eine negative Anzahl Punkte ist also nicht möglich. Den Studierenden ist eine Nummer zugewiesen (1 …). Eine Methode setPoints(int StudentNo, int points) ermöglicht das Erfassen der Punkte.
Im Zentrum dieser Aufgabe stehen die zu implementierenden Methoden getMax(), getMin() und getMean(). getMax() liefert als Ergebnis die maximal erreichte Punktzahl zurück; getMin() entsprechend die tiefste Punktzahl. getMean() berechnet den Mittelwert über alle Studierenden.
Die Aufgabe hat Entwicklungspotenzial: Erweitern Sie die Klasse um weitere sinnvolle Methoden. Welches ist z.B. der beste Studierende? Berücksichtigen Sie die Situation, falls nicht alle Studierenden den Test gemacht haben.

Notizen aus dem Lernteam-Unterricht der SW6

Vorgehensweise bei der Problemlösung in OOP:
→ Identifizierung der verschiedenen Objekte aus der realen Welt.
→ Erfassen der Verhaltensarten (Methoden) und Eigenschaften (Attribute) von Objekten.
→ Kommunikationsverhalten zwischen Objekten erfassen.

Modularität in objektorientierten Systemen:
Aufteilung eines Systems in Module
→ Modulgeschlossenheit
→ Minimalität der Schnittstellen
→ Modulkopplung
→ Modulbindung

Frage aus dem Unterricht
Was ergibt die folgende Kondition: (false || true && false)
→ AND (&&) hat die höhere Priorität als OR (||), deshalb ergibt die Kondition „false“ als Resultat.