Programmierung für Mathematik HS12
Übung 3

14. Oktober 2012

1 Aufgabe: Code verstehen

1.1 Ziele

- Präzedenz verstehen.
- If else kennen lernen.

1.2 Aufgabenstellung

1.2.1 Präzedenz

Lesen Sie folgende Codeaussschnitte. Versuchen Sie herauszufinden, welchen Wert die Variabeln am Ende haben, bevor Sie dies mit einem Programm überprüfen. Gehen Sie davon aus, dass die Codeauschnitte in einer main-Methode stehen.

Ausschnitt 1

```
int a=9,b=12;

a+++b;
```

Ausschnitt 2

```
int a=1,b=2;

b+a++;
```

Ausschnitt 3

```
int a=1,b=2;

b+ ++a;
```

Ausschnitt 4

```
int a=1,b=2;

b+ ++a+b*b;
```

Ausschnitt 5

```
int a=1,b=2;
b++*--a;
```

Ausschnitt 6

```
int a=4,b=8;

a+++b*a/--b+b*a-b--
```

Ausschnitt 7

```
int a=4,b=8;
a=a++;

a+1/b+++a-b+++b;
```

Formen Sie die Codeausschnitte um und fügen Sie Klammern ein, um die Präzedenzen aufzuzeigen.

1.3 If else Statements

Sei `a` eine boolsche Variabel, `s` einen `String`, `i` einen Integer und `f` einen Float welche der folgenden Terme dürfen in einer `If` Bedingung vorkommen?

- `f=i`

- `f==++i`
- `a`
- `a==(i==f)`
- `++a`
- `a=true`
- `a==false`

Lesen Sie folgenden Codeausschnitt aufmerksam durch. Entscheiden Sie, was auf der Konsole ausgegeben wird, wenn diese in einer main-Methode stehen.

`else if` Verschachtelung

```
int a=4,b=5,c=9;
        if(a==--b){
            System.out.println("Konz");
            if(b!=a++){
                System.out.println("ert");
            }
                else{if(c==a+b)
{System.out.println("entration");
                }else{System.out.println("4");
                }}

                }else
        {System.out.println("2");
        }
```

2 Aufgabe: Klassen schreiben

2.1 Ziele

- Erfahrung sammeln im Schreiben von Klassen
- Mathematische Konstrukte in Java umsetzen

2.2 Aufgabenstellung

In dieser Aufgabe sollen Sie mehrere Klassen schreiben. Die Klassen sollten sie fortlaufend überprüfen, damit Sie mögliche Fehler gleich entdecken.

2.2.1 Vektor

Schreiben Sie eine Klasse `Vektor`. Diese Klasse soll einen zwei-dimensionalen $\mathbb{R}$-Vektor darstellen. Dazu müssen zwei `floats` gespeichert werden. Diese Variabeln können mit Setter-Methoden verändert werden. Erweitern Sie die so erstellte Klasse um folgende Methoden:

- `add`: Welche einen `Vektor` als Input nimmt und diesen mit dem intern gespeicherten Vektor addiert, so dass nach dem Methoden-Aufruf der intern gespeicherte Vektor ersetzt wurde und nun die Summe aus den beiden `Vektoren` repräsentiert. Die Methode soll die Summe auch gleich als `Vektor` zurückgeben.
- `scalarProdukt`: Welche einen `Vektor` als Input erwartet und das Skalarprodukt des Inputvektors mit dem internen Vektor als `float` returniert.
- `norm`: Welche die Norm des Vektors returnieren soll und keinen Parameter verlangt. Dazu brauchen Sie Methode aus der Klasse `Math`. Lesen Sie in der API nach was für Methoden diese Klasse anbietet.

Hinweis: Zum Beispiel würde die Signatur der Methode `add` folgendermassen aussehen:
`public Vektor add(Vektor input)`

2.2.2 Matritzen

Schreiben Sie eine Klasse `Matrix`. Diese soll eine $2x2$ $\mathbb{R}$-Matrix, also vier `floats`, speichern können. Auf diese vier Werte soll über getter-und setter-Methoden zugegriffen werden. Erweitern Sie die Klasse um folgende Methoden:

- `det`: Welche die Determinante als float returnieren soll.
- `add`: Welche eine weitere Matrix als Parameter erwartet und die Summe der beiden Matrizen als Matrix returniert ohne auch nur eine der beiden Matrizen zu ändern.
- `act`: Welche einen `Vektor` als Parameter erwartet und einen `Vektor` returniert. Der returnierte `Vektor` soll das Produkt vom übergebenen `Vektor` und der Matrix sein.

2.2.3 Text ausgaben

Erweiteren Sie die Klasse `Vektor` um die Methode `quadrant`. Diese Methode soll eine `void` Methode sein und soll in der Konsole ausgeben in welchem Quadranten sich der Vektor befindet (d.h. ob er sich im rechten oberen Quadrant befindet oder im linken oberen Quadrant, u.s.w.). Benutzen Sie dazu die `signum` Methode der Klasse `Math` und entscheiden Sie mit `if`-und `else`-Statements welcher Text in der Konsole ausgegeben werden soll.

Schreiben Sie je eine `print` Methode für die Klasse `Matrix` und eine für `Vektor`. Die folgende Ausgabe sollte beispielsweise auf der Konsole erscheinen:

```
Matrix


------------
|2       3 |
|4       1 |
------------

Vektor

/1\
\2/
```

Wobei die Zahlen den internen Daten des Objektes entsprechen. Versuchen Sie dabei nur einen Aufruf der Methode `println` zu verwenden.