Prof. Dr. Stefan Brass
Dipl.-Inform. Annett Thüring
Dipl.-Inform. Steffen Schiele

Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik
Wintersemester 2013/ 2014

# Übungsblatt 5: Objektorientierte Programmierung

Ausgabe: 15.11.2013 Abgabe: 22.11.2013

## Aufgabe 1: Vollkommene Zahl (6 Punkte)

Schreiben Sie ein Programm, welches eine ganze positive Zahl einliest und

- alle Teiler, abgesehen von der Zahl selbst, bestimmt und diese auch ausgibt,
- angibt, ob es sich um eine abundante Zahl (Summe der Teiler ohne die Zahl selbst ist größer als die Zahl) handelt,
- angibt, ob es sich um eine defiziente Zahl (Summe der Teiler ohne die Zahl selbst ist kleiner als die Zahl) handelt und
- angibt, ob es sich um eine vollkommene Zahl (Summer der Teiler ohne die Zahl selbst ist gleich der Zahl) handelt.

*Beispiel:* Die Zahl $28$ hat die Teiler $1$, $2$, $4$, $7$ und $14$ ($28$ wird hier nicht berücksichtigt). Da die Summer dieser Teiler gleich $28$ ist, handelt es sich um eine vollkommene Zahl.

## Aufgabe 2: Arrays (4 Punkte)

Schreiben Sie ein Programm, welches zuerst die Anzahl der folgenden Eingaben einliest und entsprechend dieser Anzahl viele Eingaben von dem Benutzer in ein Array speichert. Gehen Sie davon aus, dass es sich bei diesen Eingaben um Zeichenketten handelt! Anschließend soll Ihr Programm die eingelesen Zeichenketten in umgekehrter Reihenfolge ausgeben (d. h. die letzte Eingabe wird zuerst und die erste Eingabe als letztes ausgegeben).

*Hinweis:* Bedenken Sie, dass die Methode `nextInt()` der Scanner-Klasse nur die eingegeben Zahl einliest. Wenn Sie direkt danach die Methode `nextLine()` verwenden, um eine Zeile (Zeichenkette) einzulesen, wird die Methode nur `\n` (Zeilenumbruch; siehe Vorlesung Kapitel 5, Folie 54) von der vorherigen Eingabe zurückgeben. Um nach `nextInt()` eine Zeichenkette einzulesen, müssen Sie ein zusätzliches `nextLine()` ausführen.

## Aufgabe 3: Lauflängencodierung (ohne Punkte)

Schreiben Sie ein Programm, welches einen String einliest und den mittels der Lauflängencodierung komprimierten String auf der Konsole ausgibt. Sie dürfen davon ausgehen, dass die eingegebene Zeichenkette nur Kleinbuchstaben beinhaltet. Bei einer Eingabe von
`aaabbccccccabbbcbccc`
wird die folgende Ausgabe erwartet:
`a3bbc7ab3cbc3`

*Hinweis zur Lauflängencodierung*: Kommt ein Buchstabe hintereinanderweg mehr als zweimal vor, so wird er nur einmal gefolgt von der Anzahl seines Auftretens aufgeschrieben.

**Aufgabe 4: Logische Ausdrücke (keine Abgabe)**

Die verwendeten Variablen sind wie folgt deklariert: `boolean a, b, c; int n, m;`

1. Wandeln Sie den folgenden Anweisungsblock so um, dass Sie mit genau einer `if`-Bedingung und einem `&&` auskommen:

```
if(a)
{   if(b)
    {  n=1;  }
}
```

2. Wandeln Sie den folgenden Anweisungsblock so um, dass Sie mit genau einer `if`-Bedingung auskommen:

```
if(a && b)
{   n=2;   }
if(b && c)
{   n=2;   }
```

3. Vereinfachen Sie die folgende bedingte Verzweigung soweit wie möglich:

```
if(a)
{   n=3;
    m=2;
}
else
{   n=3;
    m=2;
}
```

4. Vereinfachen Sie das folgende Programmstück soweit wie möglich:

```
if(a)
{   n=4;   }
if(!a)
{   n=7;   }
```

5. Schreiben Sie die (bezüglich Java) syntaktisch korrekte bedingte Verzweigung zu dem folgenden Satz auf: „Wenn a wahr ist oder n größer als 10 ist, dann führe `n=5;` aus, ansonsten `n=10;`“

6. Schreiben Sie die (bezüglich Java) syntaktisch korrekte bedingte Verzweigung zu dem folgenden Satz auf: „Wenn a nicht wahr ist, dann führe `n=1;` aus, ansonsten führe `n=10;` aus, falls n ungleich 10 ist, andernfalls führe `n=0;` aus.“