Prof. Dr. M. Müller C. Aschermann, J. Hensel, J. Protze, P. Reble

Allgemeine Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je **3 Studierenden** aus der **gleichen Kleingruppenübung (Tutorium)** bearbeitet werden. **Namen und Matrikelnummern** der Studierenden sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter!**
- Die **Nummer der Übungsgruppe** muss **links oben** auf das **erste Blatt** der Abgabe geschrieben werden. Notieren Sie die Gruppennummer gut sichtbar, damit wir besser sortieren können.
- Die Lösungen müssen **bis Montag, den 9.11.2014 um 15:00 Uhr** in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55). Alternativ können Sie die Lösungen auch vor der Abgabefrist direkt bei Ihrer Tutorin/Ihrem Tutor abgeben.
- In einigen Aufgaben müssen Sie in Java programmieren und `.java`-Dateien anlegen. **Drucken** Sie diese aus **und** schicken Sie sie per **E-Mail** vor Montag, dem 9.11.2014 um 15:00 Uhr an Ihre Tutorin/Ihren Tutor.
  Stellen Sie sicher, dass Ihr Programm von `javac` **akzeptiert** wird, ansonsten werden keine Punkte vergeben.

Tutoraufgabe 1 (Typen):

Bestimmen Sie den Typ und das Ergebnis der folgenden Java-Ausdrücke und begründen Sie Ihre Antwort. Sollte der Ausdruck nicht typkorrekt sein, begründen Sie, worin der Fehler besteht.
Dabei seien die Variablen `x`, `y` und `z` wie folgt deklariert: `int x = 1; int y = 2; int z = 3;`

a) `false && true`

b) `10 / 3`

c) `10 / 3.`

d) `x == y ? x > y : y < z`

e) `(byte) (127 + 1)`

f) `'x' + y + z`

g) `x + y + "z"`

h) `1 || 0`

Aufgabe 2 (Typen): (4 Punkte)

Bestimmen Sie den Typ und das Ergebnis der folgenden Java-Ausdrücke und begründen Sie Ihre Antwort. Sollte der Ausdruck nicht typkorrekt sein, begründen Sie, worin der Fehler besteht.
Dabei seien die Variablen `x`, `y` und `z` wie folgt deklariert: `int x = 1; int y = 2; int z = 122;`

a) `2147483600 + '2'`

b) `(double) (3 / 2)`

c) `"x" + y + z`

d) `x + y > z && true`

e) `9 / 4.5`

f) `"x" - z`

g) `17 + 0f`

h) `z != 'z'? 1f : 2`

Tutoraufgabe 3 (Zweierkomplement):

a) Erklären Sie im Detail, wie die beiden Ausgaben des folgenden Programms berechnet werden.

```
public class Test {
    public static void main(String[] args) {
        int zahl = -2147483648;

        System.out.println(zahl + 1);
        System.out.println(zahl - 1);
    }
}
```

Hinweis: $-2^{31} = -2147483648$

b) Welche Zahlen repräsentieren die folgenden Bitfolgen im 5-Bit Zweierkomplement?

00010 10111 11011 01101 10000

c) Sei x eine ganze Zahl. Wie unterscheiden sich die Zweierkomplement-Darstellungen von x und $-x$?

Aufgabe 4 (Zweierkomplement): (2+2 Punkte)

a) Welche Zahlen repräsentieren die folgenden Bitfolgen im 10-Bit Zweierkomplement?

0100000101 1111010110 0001011011 1000010000

b) Die zwei folgenden Java-Ausdrücke werten jeweils zu `true` aus, obwohl dies auf den ersten Blick nicht offensichtlich erscheint. Geben Sie dafür jeweils eine kurze informelle Begründung an.

i) $2\,000\,000\,000 + 1\,000\,000\,000 < 2\,000\,000\,000$

ii) $-(-2\,147\,483\,648) == -2\,147\,483\,648$

Tutoraufgabe 5 (Input):

Schreiben Sie ein einfaches Java-Programm, welches den Benutzer auffordert, eine positive Zahl ganze Zahl (d. h. größer als 0) einzugeben. Danach soll das Programm eine durch die Return/Enter-Taste beendete Zahl einlesen. Diese Eingabeaufforderung mit anschließendem Einlesen soll solange wiederholt werden, bis der Benutzer eine positive Zahl eingibt. Anschließend soll der Benutzer aufgefordert werden, ein Wort einzugeben. Dieses soll ebenfalls durch die Return/Enter-Taste beendet werden. Das Wort soll eingelesen und schließlich so oft hintereinander in einer Zeile ausgeben werden, wie durch die eingegebene positive Zahl festgelegt wurde.

Aufgabe 6 (Arithmetisches Mittel): (5 Punkte)

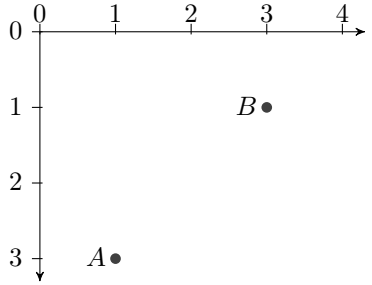
Schreiben Sie ein einfaches Java-Programm zur Berechnung des arithmetischen Mittels einer Menge ganzer Zahlen. Der Benutzer soll zuerst nach der Anzahl der Werte, von denen das Mittel berechnet werden soll, gefragt werden. Alle Eingaben des Benutzers sollen mit der Return/Enter-Taste beendet werden. Die Aufforderung zur Eingabe soll so lange wiederholt werden, bis der Benutzer eine positive Zahl ≥ 1 eingibt. Anschließend sollen entsprechend viele Integer-Werte eingelesen werden, deren arithmetisches Mittel am Ende ausgegeben wird.

Die Ausgabe eines beispielhaften Ablaufs des Programms könnte wie folgt aussehen:

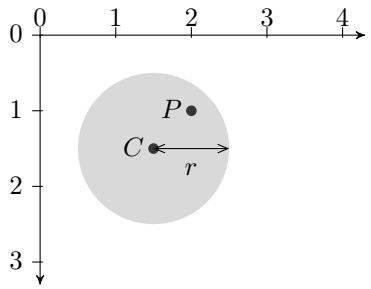
```
Bitte geben Sie die Anzahl der Werte an, fuer die Sie das
arithmetische Mittel berechnen wollen: 0
Die eingegebene Zahl ist nicht positiv!
Bitte geben Sie die Anzahl der Werte an, fuer die Sie das
arithmetische Mittel berechnen wollen: 3
Bitte geben Sie Wert 1 ein:
17
Bitte geben Sie Wert 2 ein:
4
Bitte geben Sie Wert 3 ein:
11
Das arithmetische Mittel betraegt 10.666666666666666
```

Tutoraufgabe 7 (Das Herz):

In dieser Aufgabe sollen Sie ein Programm schreiben, das auf der Konsole ein Herz ausgibt. Im Folgenden betrachten wir ein Koordinatensystem, in dem der Ursprung links oben ist. Betrachten Sie zur Illustration das folgende Bild, in dem der Punkt A an $(1, 3)$ und der Punkt B an $(3, 1)$ liegt:



a) Wir wollen nun untersuchen, ob ein Punkt P mit den Koordinaten (p_x, p_y) in einem Kreis liegt, den wir mit Hilfe eines Punktes C (an Position (c_x, c_y)) und eines Radius r angeben. Dies wird im folgenden Schaubild für $P = (2, 1)$, $C = (1.5, 1.5)$ und $r = 1$ illustriert:



Ein Punkt (p_x, p_y) liegt dann in einem Kreis, wenn sein Abstand vom Mittelpunkt (c_x, c_y) kleiner oder gleich dem Radius r ist (d. h. Punkte auf dem Rand des Kreises zählen als innerhalb des Kreises liegend).

Vervollständigen Sie nun die folgende Funktion so, dass `true` zurückgegeben wird, wenn (`px`, `py`) im vom Mittelpunkt (`cx`, `cy`) und Radius `r` definierten Kreis liegt und `false` in allen anderen Fällen:

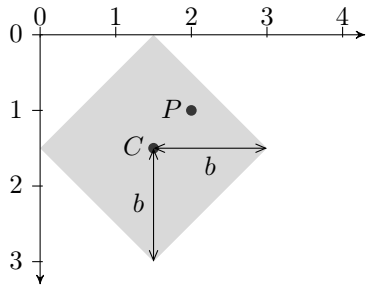
```
public static boolean inKreis(double px, double py, double cx,
                              double cy, double r) {
  return...
}
```

Sie dürfen dabei annehmen, dass der Radius `r` eine positive Zahl ist.

Hinweise:

- Die Anweisung "`return` *expr*" wertet den Ausdruck *expr* aus und setzt das Ergebnis an der Aufrufstelle der Funktion ein.
- Verwenden Sie die Funktion `Math.sqrt(double a)`, die $\sqrt[2]{\texttt{a}}$ als `double` zurückgibt.

b) Wir wollen nun untersuchen, ob ein Punkt P mit den Koordinaten (p_x, p_y) in einer quadratischen Raute liegt, die wir mit Hilfe eines Punktes C (an Position (c_x, c_y)) und einer Rautenbreite b definieren. Dies wird im folgenden Schaubild für $P = (2, 1)$, $C = (1.5, 1.5)$ und $b = 1.5$ illustriert:



Ein Punkt (p_x, p_y) liegt dann in einer Raute um (c_x, c_y), wenn die Summe der Differenzen $|p_x - c_x| + |p_y - c_y|$ kleiner oder gleich der Rautenbreite b ist (d. h. Punkte auf dem Rand der Raute zählen als innerhalb der Raute liegend).

Vervollständigen Sie nun die folgende Funktion so, dass `true` zurückgegeben wird, wenn (`px`, `py`) in der Raute um den Mittelpunkt (`cx`, `cy`) und Rautenbreite `b` liegt und `false` in allen anderen Fällen:

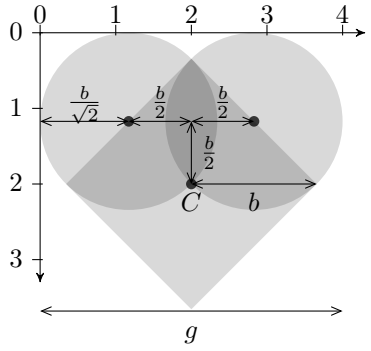
```
public static boolean inRaute(double px, double py, double cx,
                              double cy, double b) {
  return ...
}
```

Sie dürfen dabei annehmen, dass die Rautenbreite `b` eine positive Zahl ist.

Hinweise:

- Verwenden Sie die Funktion `Math.abs(double a)`, die $|\texttt{a}|$ als `double` zurückgibt.

c) Wir wollen nun untersuchen, ob ein Punkt in einem Herz liegt. Betrachten Sie zuerst das folgende Schaubild:



Ein Punkt liegt dann in einem Herz mit Gesamtbreite g und Mittelpunkt $C = (c_x, c_y)$, wenn er in einer Raute mit Rautenbreite $b = \frac{g}{1+\frac{2}{\sqrt{2}}}$ um C, in einem Kreis mit Radius $\frac{b}{\sqrt{2}}$ um $(c_x - \frac{b}{2}, c_y - \frac{b}{2})$ oder in einem Kreis mit dem gleichen Radius um $(c_x + \frac{b}{2}, c_y - \frac{b}{2})$ liegt. Im Beispiel ist $g = 4$ gewählt worden.

Verwenden Sie nun die Funktionen `inKreis` aus Teilaufgabe a) und `inRaute` aus Teilaufgabe b), um die folgende Funktion zu implementieren, die genau dann `true` zurückgibt, wenn ein Punkt (`px`, `py`) in einem durch den Mittelpunkt (`cx`, `cy`) und die Gesamtbreite `g` beschriebenen Herzen liegt:

```
public static boolean inHerz(double px, double py, double cx,
                             double cy, double g) {
  return ...
}
```

Sie dürfen dabei annehmen, dass die Gesamtbreite `g` eine positive Zahl ist.

Hinweise:

- Verwenden Sie die boolesche Verknüpfung `||`, um die verschiedenen Bedingungen miteinander zu verbinden.

d) Sie sollen nun die Funktion `inHerz` aus Aufgabenteil c) verwenden, um ein Herz auf der Konsole auszugeben.

Schreiben Sie nun ein Programm, das

1. den Benutzer nach der Gesamtbreite g für das Herz fragt und eine Integer-Zahl einliest. Bestimmen Sie daraus $(\frac{g}{2}, \frac{g}{2})$ als Mittelpunkt für das Herz. Zur Vereinfachung dürfen Sie annehmen, dass der Benutzer hier eine positive ganze Zahl eingibt.
2. mit Hilfe zweier Schleifen alle Koordinaten $(0,0), \ldots, (0,g), (1,0), \ldots (1,g), \ldots (g,g)$ abläuft und ein "`#`" ausgibt, wenn sich die Koordinate im Herz befindet und sonst " " ausgibt.

Ein Aufruf des Programms soll dann etwa das folgende Resultat zeigen:

```
Geben Sie die gewuenschte Gesamtbreite ein: 30

     #########    #########
   ##########################
  ############################
 ##############################
 ##############################
  ############################
   ##########################
    ########################
     ######################
       ##################
         ##############
           ##########
             #######
               ###
```

Hinweise:

- Wenn Sie mit dem Aussehen Ihres Herzens unzufrieden sind, weil es vertikal gestreckt ist, liegt dies vermutlich daran, dass in üblichen Schriftarten ein Zeichen ca. 2 mal so hoch wie breit ist. Wollen Sie ein gleichmäßigeres Ergebnis erzielen, so dürfen Sie die y-Werte (also die Höhe) aller Koordinaten vor der Eingabe in `inHerz` mit 2 multiplizieren. Dies wurde auch bei der obigen Beispielausgabe gemacht.
- Sie dürfen statt eines "#" für ein schöneres Ergebnis auch das Unicode-Zeichen "\u2665" (dargestellt als ♥) verwenden.

Aufgabe 8 (Das Haus): (1 + 3 + 5 + 2 = 11 Punkte)

In dieser Aufgabe sollen Sie ein Programm schreiben, das auf der Konsole ein Haus ausgibt. Die zur Illustration verwendeten Koordinatensysteme werden in der vorhergehenden Tutoraufgabe eingehend erklärt.

a) Wir wollen zuerst untersuchen, ob ein Punkt p mit den Koordinaten (p_x, p_y) in einem Rechteck liegt, das später den Hauptteil des Hauses darstellen soll.

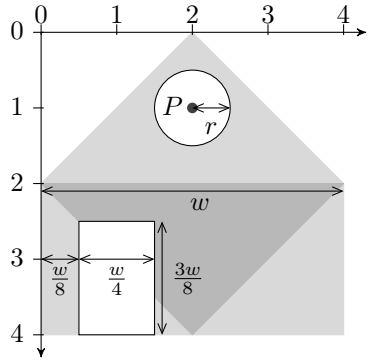
Vervollständigen Sie nun die folgende Funktion so, dass `true` zurückgegeben wird, wenn $(\texttt{px}, \texttt{py})$ in dem durch die obere linke Ecke $ol = (\texttt{olx}, \texttt{oly})$ und die untere rechte Ecke $ur = (\texttt{urx}, \texttt{ury})$ definierten Rechteck liegt und `false` in allen anderen Fällen:

```
public static boolean inRechteck(double px, double py, double olx,
                                  double oly, double urx, double ury)
{
  return...
}
```

Hinweise:

- Die Anweisung "`return` *expr*" wertet den Ausdruck *expr* aus und setzt das Ergebnis an der Aufrufstelle der Funktion ein.

b) Wir wollen nun untersuchen, ob ein Punkt in einem Haus der Breite w liegt. Das fertige Haus wird im folgenden Schaubild für $w = 4$, $r = w/8$, und $P = (w/2, w/4)$ illustriert:



Ein Punkt liegt genau dann in dem Haus mit Gesamtbreite w, wenn er gleichzeitig die folgenden Eigenschaften erfüllt:

- Er liegt in einem Rechteck mit den Parametern $ol = (0, w/2)$, $ur = (w, w)$ oder einer quadratischen Raute mit Zentrum $C = (w/2, w/2)$ und Rautenbreite $b = w/2$.
- Er liegt **nicht** im Fenster des Hauses, das durch einen Kreis mit Mittelpunkt $(w/2, w/4)$ und Radius $r = w/8$ dargestellt wird.
- Er liegt **nicht** in der Tür des Hauses. Diese wird durch ein Rechteck mit den Parametern $ol = (w/8, 5w/8)$ und $ur = (3w/8, w)$ beschrieben.

Verwenden Sie nun die Funktionen `inRaute` und `inKreis` aus der vorherigen Tutoraufgabe und `inRechteck` aus Teilaufgabe a), um die folgende Funktion zu implementieren, die genau dann `true` zurückgibt, wenn ein Punkt $(\texttt{px}, \texttt{py})$ in einem durch die Gesamtbreite `w` beschriebenen Haus liegt:

```
public static boolean inHaus(
        double px,
        double py,
        double w)
{
  return ...
}
```

Sie dürfen dabei annehmen, dass die Gesamtbreite `w` eine positive Zahl ist.

Hinweise:

- Verwenden Sie boolesche Verknüpfungen (`||` und/oder `&&`) sowie die Negation (`!`), um die verschiedenen Bedingungen miteinander zu verbinden.

c) Sie sollen nun die Funktion `inHaus` aus Aufgabenteil b) verwenden, um ein Haus auf der Konsole auszugeben.

Schreiben Sie ein Programm, das

1. den Benutzer nach der Gesamtbreite w für das Haus fragt und eine Integer-Zahl einliest. Zur Vereinfachung dürfen Sie annehmen, dass der Benutzer hier eine positive ganze Zahl eingibt.
2. mit Hilfe zweier Schleifen alle Koordinaten $(0,0),\ldots,(0,w),(1,0),\ldots(1,w),\ldots(w,w)$ abläuft und ein "`#`" ausgibt, wenn sich die Koordinate im Haus befindet, und sonst " " ausgibt.

Ein Aufruf des Programms soll dann etwa das folgende Resultat zeigen:

```
Geben Sie die gewuenschte Gesamtbreite ein: 46
                        *
                       *****
                      *********
                    *****   *****
                  ****         ****
                ******          ******
              *******            *******
            *********            *********
          *************      *************
       ***************************************
     *****************************************
   *********************************************
 ************************************************
**************************************************
**************************************************
**************************************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
******              ******************************
```

Hinweise:

- Wenn Sie mit dem Aussehen Ihres Hauses unzufrieden sind, weil es vertikal gestreckt ist, liegt dies vermutlich daran, dass in üblichen Schriftarten ein Zeichen ca. 2 mal so hoch wie breit ist. Wollen Sie ein gleichmäßigeres Ergebnis erzielen, so dürfen Sie die y-Werte (also die Höhe) aller Koordinaten vor der Eingabe in `inHaus` mit 2 multiplizieren. Dies wurde auch bei der obigen Beispielausgabe gemacht.

d) Passen Sie das Programm so an, dass die Aufforderung zur Eingabe einer Gesamtbreite so lange wiederholt wird, bis die vom Benutzer eingegebene Zahl mindestens 20 beträgt. Bieten Sie ihm außerdem die Möglichkeit, nach Eingabe der Breite eines von drei möglichen Zeichen zu wählen, aus dem das Haus bestehen soll. Eine mögliche Ausgabe bis zu diesem Punkt könnte aussehen wie folgt:

```
Geben Sie die gewuenschte Gesamtbreite ein (mind. 20): 10
Geben Sie die gewuenschte Gesamtbreite ein (mind. 20): 45
```

```
Aus welchem Zeichen soll das Haus bestehen?
1: Plus (+)
2: Stern (*)
3: Herz (♥)
```

Hier kann der Benutzer mit "1", "2" oder "3" sein bevorzugtes Zeichen wählen. Nutzen Sie eine `switch`-Anweisung zur Ausgabe der korrekten Zeichen. Falls der Benutzer eine ungültige Zahl eingibt, soll hier "`#`" gewählt werden.

Hinweise:

- Um ein Herz darzustellen, können Sie das Unicode-Zeichen "\u2665" verwenden.