

Lösungsvorschlag zur 5. Übung zur Vorlesung
Einführung in die Programmierung

Hinweise zu den Hausübungen Bitte achten Sie darauf, in sämtlichen abgegebenen **.java**-Dateien jegliche Zeilen mit **package** zu entfernen (wird oft von IDEs am Anfang eingefügt). **package** Angaben in abgegebenen **.java**-Dateien können ab sofort zu Punktabzug führen, da dies den Zeitaufwand für die Korrektur unnötig erhöht! Das Thema „Packages“ wird in der Vorlesung später noch behandelt werden. Es werden nur **.java**-Dateien korrigiert, die kompilieren! Kommentieren Sie problematischen Code aus und schreiben Sie Fragen dazu.

Achten Sie ebenfalls darauf, dass die Dateinamen keine Umlaute enthalten, da UniWorX momentan leider Probleme mit Umlauten in Dateinamen hat.

A5-1 BNF Ein Text ist eine nicht-leere Sequenz von Sätzen. Ein Satz besteht aus einem Subjekt, gefolgt von einem Prädikat und einem Objekt. Jeder Satz wird durch einen Punkt abgeschlossen. Das Objekt kann ein Wort oder ein in (französischen) Anführungszeichen (\gg bzw. \ll) eingeschlossener Text sein.

Als Hilfe sind in dieser Aufgabe alle Terminalsymbole mit deutschen Anführungszeichen (\ll bzw. \gg) gekennzeichnet. Die deutschen Anführungszeichen sind kein Teil der Sprache.

- a) Geben Sie eine BNF-Grammatik für die oben beschriebenen Texte an. Dabei können Sie folgende Regeln als bereits gegeben betrachten:

$$\begin{aligned}\langle \text{Subjekt} \rangle &:= \text{“Hund”} \mid \text{“Katze”} \\ \langle \text{Prädikat} \rangle &:= \text{“sagt”} \mid \text{“brummt”} \\ \langle \text{Wort} \rangle &:= \text{“wuff”} \mid \text{“miau”} \\ \langle \text{Punkt} \rangle &:= \text{“.”} \\ \langle \text{AnführungLinks} \rangle &:= \text{“}\gg\text{”} \\ \langle \text{AnführungRechts} \rangle &:= \text{“}\ll\text{”}\end{aligned}$$

Zur Erinnerung: Eine BNF-Grammatik ist ein Quadrupel (Σ, V, S, P) , siehe Folie 5.11.

LÖSUNGSVORSCHLAG:

Wir (Σ, V, S, P) mit

- $\Sigma = \{ \text{“Hund”}, \text{“Katze”}, \text{“sagt”}, \text{“brummt”}, \text{“wuff”}, \text{“miau”}, \text{“.”}, \text{“}\gg\text{”}, \text{“}\ll\text{”} \}$
- $V = \{ \langle \text{Text} \rangle, \langle \text{Satz} \rangle, \langle \text{Objekt} \rangle, \langle \text{Subjekt} \rangle, \langle \text{Prädikat} \rangle, \langle \text{Wort} \rangle, \langle \text{Punkt} \rangle, \langle \text{AnführungLinks} \rangle, \langle \text{AnführungRechts} \rangle \}$
- $S = \{ \langle \text{Text} \rangle \}$
- Die Menge P besteht aus folgenden Produktionen:
$$\begin{aligned}\langle \text{Text} \rangle &:= (\langle \text{Satz} \rangle)^+ \\ \langle \text{Satz} \rangle &:= \langle \text{Subjekt} \rangle \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \langle \text{Punkt} \rangle \\ \langle \text{Objekt} \rangle &:= \langle \text{Wort} \rangle \mid \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle\end{aligned}$$

b) Leiten Sie aus Ihrer Grammatik den folgenden Text ab:


“Hund” “brummt” “»” “Katze” “sagt” “miau” “.” “«” “.”

LÖSUNGSVORSCHLAG:

$\langle \text{Text} \rangle \rightarrow (\langle \text{Satz} \rangle)^+ \rightarrow \langle \text{Satz} \rangle \rightarrow \langle \text{Subjekt} \rangle \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \langle \text{Punkt} \rangle$
 $\rightarrow (\text{“Hund”} | \text{“Katze”}) \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \langle \text{Punkt} \rangle$
 $\rightarrow \text{“Hund”} \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \langle \text{Punkt} \rangle \rightarrow \text{“Hund”} \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \text{“.”}$
 $\rightarrow \text{“Hund”} (\text{“sagt”} | \text{“brummt”}) \langle \text{Objekt} \rangle \text{“.”}$
 $\rightarrow \text{“Hund”} (\text{“sagt”} | \text{“brummt”}) (\langle \text{Wort} \rangle | \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle) \text{“.”}$
 $\rightarrow \text{“Hund”} (\text{“sagt”} | \text{“brummt”}) \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \langle \text{AnführungLinks} \rangle (\langle \text{Satz} \rangle)^+ \langle \text{AnführungRechts} \rangle \text{“.”}$
 $\rightarrow^2 \text{“Hund”} \text{“brummt”} \text{“»”} \langle \text{Satz} \rangle \text{“«”} \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \text{“»”} \langle \text{Subjekt} \rangle \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \langle \text{Punkt} \rangle \text{“«”} \text{“.”}$
 $\rightarrow^2 \text{“Hund”} \text{“brummt”} \text{“»”} (\text{“Hund”} | \text{“Katze”}) (\text{“sagt”} | \text{“brummt”}) \langle \text{Objekt} \rangle \text{“.”} \text{“«”} \text{“.”}$
 $\rightarrow^3 \text{“Hund”} \text{“brummt”} \text{“»”} \text{“Katze”} \text{“sagt”} (\langle \text{Wort} \rangle | \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle) \text{“.”} \text{“«”} \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \text{“»”} \text{“Katze”} \text{“sagt”} \langle \text{Wort} \rangle \text{“.”} \text{“«”} \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \text{“»”} \text{“Katze”} \text{“sagt”} (\text{“wuff”} | \text{“miau”}) \text{“.”} \text{“«”} \text{“.”}$
 $\rightarrow \text{“Hund”} \text{“brummt”} \text{“»”} \text{“Katze”} \text{“sagt”} \text{“miau”} \text{“.”} \text{“«”} \text{“.”}$

Die Reihenfolge, in der wir die Produktionen anwenden, ist beliebig; es gibt daher viele Lösungsmöglichkeiten. Die Reihenfolge dieser Lösung wurde gewählt um dies zu verdeutlichen.

An den mit \rightarrow^2 markierten Stellen wurden zwei Schritte auf einmal gemacht, bei \rightarrow^3 sogar drei. Für Hausaufgaben und Klausur sind solche Zusammenfassungen nur dann erlaubt, wenn es wirklich parallele Schritte sind, d.h. jedes neue Nonterminal muss einmal hingeschrieben werden, bevor man es wieder ersetzt, da ansonsten eine Korrektur nur schwer möglich ist. Lösungen mit sequentiellen Zusammenfassungen werden nicht akzeptiert, wie z.B.


 $\langle \text{Text} \rangle \rightarrow \langle \text{Subjekt} \rangle \langle \text{Prädikat} \rangle \langle \text{AnführungLinks} \rangle \langle \text{Text} \rangle \langle \text{AnführungRechts} \rangle \langle \text{Punkt} \rangle$
 $\rightarrow \text{“Hund”} \text{“brummt”} \text{“»”} \text{“Katze”} \text{“sagt”} \text{“miau”} \text{“.”} \text{“«”} \text{“.”}$

A5-2 Die Klasse *GraphicsWindow* Prof. Hofmann hat Ihnen über die Vorlesungshomepage freundlicherweise sein Klasse `GraphicsWindow` zur Verfügung gestellt. In dieser Aufgabe wollen wir verstehen, wie wir die Klasse `GraphicsWindow` einsetzen können, auch wenn wir den Inhalt der Klassendefinition mit den bis jetzt in der Vorlesung behandelten Themen noch nicht verstehen können.

- a) Speichern Sie die drei Dateien `GraphicsWindow.java`, `Car.java` und `Test.java` in ein frisches Verzeichnis. Kompilieren Sie die Klasse `Test` und starten Sie deren `main`-Methode anschließend (ggf. auch mit IDE). Was passiert?

Hinweis: Wer nicht weiß, was zu tun ist, sollte sich noch einmal Aufgabe A1-1 anschauen!

LÖSUNGSVORSCHLAG:

```
> javac Test.java
> java Test
Ein Fenster wurde geöffnet. Bitte Fenster beachten!
Fenster und Programm wurden nun beendet.
```

- b) Welche Konstruktoren und Methoden bietet die Klasse `GraphicsWindow` an? Lösen Sie diese Aufgabe mit Hilfe von Javadoc (ggf. auch mit IDE).

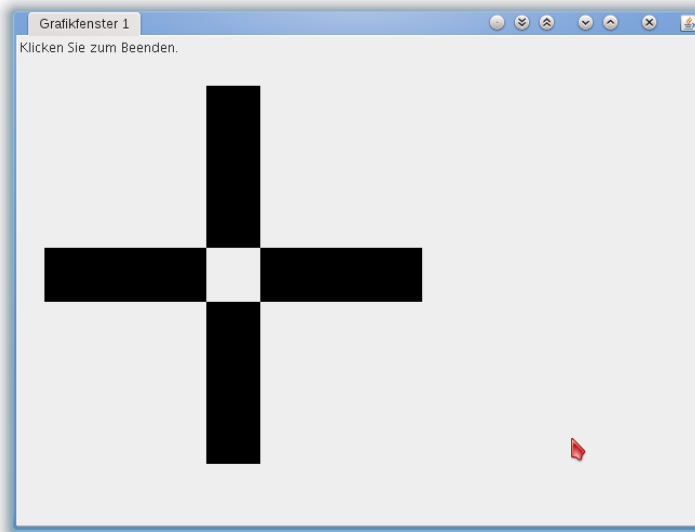
Hinweis: Da die Dateivorlage in UTF-8 kodiert ist und deutsche Umlaute erhält, benötigt Javadoc ggf. die Parameter `-encoding UTF-8 -charset UTF-8 -docencoding UTF-8`.

LÖSUNGSVORSCHLAG:

Wir rufen `javadoc GraphicsWindow.java -encoding UTF-8 -charset UTF-8 -docencoding UTF-8` und öffnen anschließend die erstellte Datei `index.html` mit einem Browser unserer Wahl.

Wer eine IDE benutzt, kann vermutlich auch einfach direkt die Dokumentation per Klick dazu aufrufen.

- c) Schreiben Sie eine eigene Klasse `Demo` zur Demonstration der Klasse `GraphicsWindow`. Orientieren Sie sich dabei an `Test`. Ihr Demo soll ungefähr das folgende Bild zeichnen:

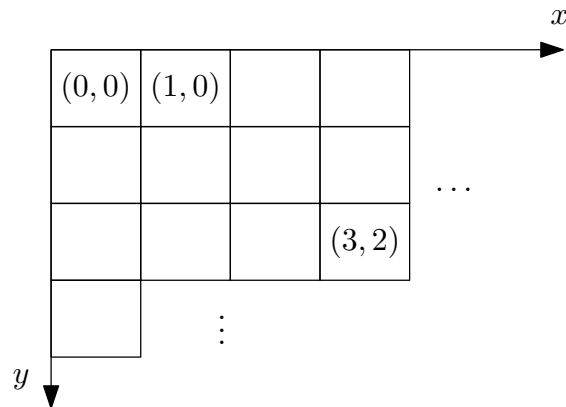


Hinweise:

- Die Ihnen bereits bekannte Klasse `java.awt.Rectangle` könnte hierfür nützlich sein. Um ein Objekt der Klasse `Rectangle` zu zeichnen, übergeben Sie es als Parameter an die Methoden `draw` oder `fill` eines Objektes der Klasse `GraphicsWindow`.
- Sie können folgendes Programmgerüst verwenden:

```
import java.awt.Rectangle;
public class Demo {
    public static void main(String[] args) {
        int margin      = 25;
        int box_height   = 50;
        int box_width    = 150;
        Rectangle r1 = new Rectangle(margin, margin+box_width,
                                    box_width, box_height);
        GraphicsWindow gw = new GraphicsWindow();
        gw.fill(r1);
        // ...
        gw.setText("Klicken Sie zum Beenden.");
        gw.mouseClick();
        System.exit(0);
    }
}
```

- Das Koordinatensystem von `GraphicsWindow` ist folgendermaßen zu verstehen:



LÖSUNGSVORSCHLAG:

Wir zeigen hier gleich 2 Lösungsmöglichkeiten in einer Datei. Bitte PFD heranzoomen.

```
import java.awt.*;

public class Demo {
    public static void main(String[] args) {
        // Eckwerte der Grafik als leicht veränderbare Variablen anlegen.
        int margin      = 25;
        int box_height   = 50;
        int box_width    = 150;

        GraphicsWindow gr = new GraphicsWindow(); // Grafikfenster öffnen
        gr.sleep(666);

        /* 1. Variante: Wir zeichnen einfach 4 Rechtecke! */
        // Die 4 Rechtecke der Grafik definieren
        Rectangle bar1 = new Rectangle(margin, margin+box_width, box_width, box_height);
        Rectangle bar2 = new Rectangle(margin+(box_height+box_width),margin+box_width, box_width, box_height);
        Rectangle bar3 = new Rectangle(margin+box_width, margin, box_height,box_width);
        Rectangle bar4 = new Rectangle(margin+box_width, margin+(box_height+box_width),box_height,box_width);
        // Zeichnen
        gr.fill(bar1);
        gr.setText("Klicken Sie zum Fortfahren.");
        gr.mouseClick(); // Mausklick Abfragen zum Schrittweisen zeichnen
        gr.fill(bar2);
        gr.mouseClick();
        gr.fill(bar3);
        gr.mouseClick();
        gr.fill(bar4);

        gr.setText("Fertig mit Variante 1. Klicken Sie zum Fortfahren.");
        gr.mouseClick();

        // Grafik wieder löschen
        gr.setText("Löschen von mit Variante 1.");
        gr.switchToBackgroundColor();
        gr.fill(bar1); gr.sleep(666); // Wir warten einfach 666ms, damit man sehen kann, was passiert.
        gr.fill(bar2); gr.sleep(666);
        gr.fill(bar3); gr.sleep(666);
        gr.fill(bar4); gr.sleep(666);
        gr.switchToForegroundColor();

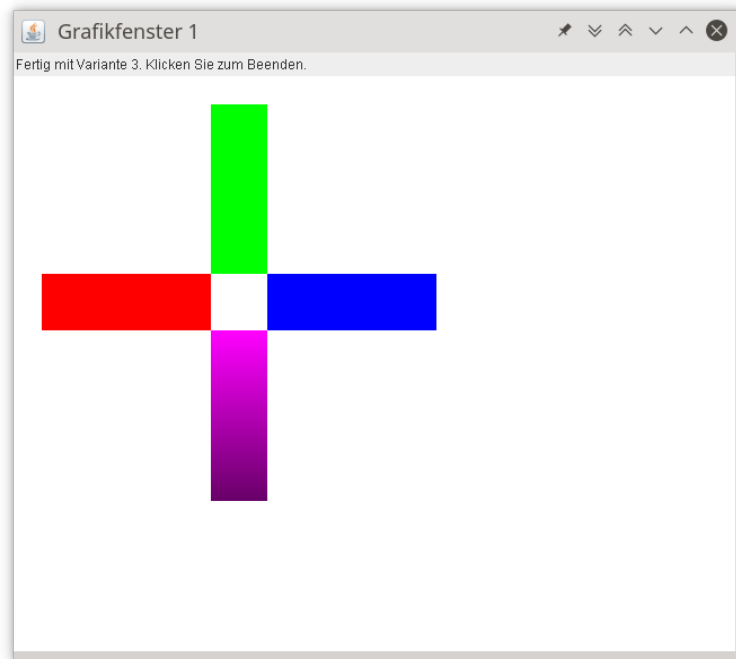
        /* 2. Variante: wir definieren lediglich 2 lange Rechtecke */
        gr.setText("Zeichnen von Variante 2.");

        Rectangle bar11 = new Rectangle(margin, margin+box_width, box_width+box_width+box_height,box_height);
        Rectangle bar12 = new Rectangle(margin+box_width,margin, box_height,box_width+box_width+box_height);

        gr.fill(bar11);
        gr.sleep(666);
        gr.fill(bar12);
        gr.sleep(666);
        gr.switchToBackgroundColor();
        gr.fill(bar12.intersection(bar11));

        gr.setText("Fertig mit Variante 2. Klicken Sie zum Beenden.");
        gr.mouseClick();
        System.exit(0);
    }
}
```

- d) Ändern Sie Ihre Lösung zu Aufgabenteil c) so ab, dass Sie jede Zeichenoperation einzeln beobachten können. Hierzu könnten die Methoden `mouseClick` und/oder `sleep` der Klasse `GraphicsWindow` nützlich sein.
- e) Ändern Sie Ihre Programm, so dass folgendes Bild entsteht:



Hinweise: Die Zeichenfarbe wird in `GraphicsWindow` mit der Methode `setColor` gesetzt. Als Argument wird ein Objekt der Klasse `java.awt.Color` erwartet. Diese bietet verschiedene Konstruktoren, z.B. `Color(int red, int green, int blue)` erwartet ganze Zahlen im Bereich 0–255 oder `Color(float r, float g, float b)` erwartet Zahlen im Bereich 0.0–1.0; den Datentyp `float` können Sie aus `double` konvertieren. In beiden Fällen gibt es eine Fehlermeldung, falls ein Argument nicht im erwarteten Zahlenbereich liegt!

LÖSUNGSVORSCHLAG:

Die oberen drei Rechtecke sind leicht einzufärben, indem wir einfach die Zeichenfarbe ändern, bevor wir das jeweilige Rechteck zeichnen.

Das untere Rechteck ist etwas kniffliger, da `GraphicsWindow` leider keine Füllung mit Schattierungen anbietet. Entweder man zeichnet das Rechteck mit einzelnen Punkten, dann brauchen wir zwei verschachtelte Schleifen, oder wir nehmen ein Rechteck der Höhe 1.

In der vorangegangenen Lösung fügen wir einfach folgendes vor `System.exit(0);` ein:

```
// Löschen des Bildes
gr.switchToBackgroundColor();
gr.clear();
// Hier geht es los:
gr.setColor(Color.RED);
gr.fill(bar1);
gr.sleep(666);
gr.setColor(Color.GREEN);
gr.fill(bar3);
gr.sleep(666);
gr.setColor(Color.BLUE);
gr.fill(bar2);
gr.sleep(666);
Rectangle bar24 = new Rectangle(bar4);
bar24.setSize(box_height,1);
for (int y=0; y<=box_width; y++) {
    Color c = new Color(255-y,0,255-y);
    gr.setColor(c);
    gr.fill(bar24);
    bar24.translate(0,1);
}
gr.setText("Fertig mit Variante 3. Klicken Sie zum Beenden.");
gr.mouseClick();
```

H5-1 *Backus-Naur Form* (6 Punkte; Abgabe: H5-1.txt oder H5-1.pdf)

Gegeben sind folgende Produktionen einer BNF-Grammatik mit dem Startsymbol $\langle Smiley \rangle$:

$$\begin{aligned}\langle Smiley \rangle &::= \langle FröhlicherSmiley \rangle \mid \langle TraurigerSmiley \rangle \\ \langle FröhlicherSmiley \rangle &::= (": " \mid "; " \mid "8") [\langle Nase \rangle] \langle FröhlicherMund \rangle \\ \langle TraurigerSmiley \rangle &::= (": " \mid "=") [\langle Nase \rangle] \langle TraurigerMund \rangle \\ \langle Nase \rangle &::= "o" \mid "-" \\ \langle FröhlicherMund \rangle &::= ")" \mid "{" ("")^+ \mid "D" \\ \langle TraurigerMund \rangle &::= "("^+ \mid "I"\end{aligned}$$

- a) Entscheiden Sie für die folgenden Wörter jeweils, ob sie in der Sprache dieser Grammatik sind. Begründen Sie Ihre Antwort in 1–3 Sätzen! Geben Sie weiterhin für Wörter, die in der Sprache sind, eine ausführliche Ableitung an! Führen Sie dabei jeden Schritt *einzeln* aus!

i) 8oI

LÖSUNGSVORSCHLAG:

Nicht in der Grammatik!

Das Terminal "8" kommt nur im Nonterminal $\langle FröhlicherSmiley \rangle$ vor, aber dieses erzwingt $\langle FröhlicherMund \rangle$; dagegen kommt das Terminal "I" nur in $\langle TraurigerMund \rangle$ vor, welches weder von $\langle Nase \rangle$ noch $\langle FröhlicherMund \rangle$ erreichbar ist. Somit kann keine Ableitung möglich sein.

ii) :-(((

LÖSUNGSVORSCHLAG:

$$\begin{aligned}
 \langle \textit{Smiley} \rangle &\rightarrow \langle \textit{FröhlicherSmiley} \rangle \mid \langle \textit{TraurigerSmiley} \rangle \rightarrow \langle \textit{TraurigerSmiley} \rangle \\
 &\rightarrow (": " \mid "=") [\langle \textit{Nase} \rangle] \langle \textit{TraurigerMund} \rangle \rightarrow ": " [\langle \textit{Nase} \rangle] \langle \textit{TraurigerMund} \rangle \\
 &\rightarrow ": " \langle \textit{Nase} \rangle \langle \textit{TraurigerMund} \rangle \\
 &\rightarrow ": " ("o" \mid "-") \langle \textit{TraurigerMund} \rangle \rightarrow ": " "-" \langle \textit{TraurigerMund} \rangle \\
 &\rightarrow ": " "-" (("(")^+ \mid "|") \rightarrow ": " "-" ("(")^+ \rightarrow ": " "-" "(" "(" "(" "(" "("
 \end{aligned}$$

b) Geben Sie einen Smiley in der Sprache dieser Grammatik an, der aus genau 5 Zeichen (Terminalsymbolen) besteht und das Terminalsymbol "(" nicht enthält!

LÖSUNGSVORSCHLAG:

$:-\{\}\}, 8o\{\})$ oder auch $;\{\})$, aber weder $:-\})$ noch $:-((($.

- LÖSUNGSVORSCHLAG:**
 $:-\{\})$, $8o\{\})$ oder auch $;\{\})$, aber weder $:-\})$ noch $:-((($.

c) Geben Sie eine Grammatik als 4-Tupel an, deren Sprache aus nicht-leeren Folgen von Smileys besteht. Dabei muss aber auf jeden traurigem Smiley mindestens zwei glückliche Smileys folgen. Hier ein paar Beispiele:

Enthalten: “;-)” , “;-) :o) :o)” , “:-| :) :) :)” oder “=(((:) :) :) =(:) ;)”

Nicht enthalten: “:-|” , “;) :-| :)” , oder “:-| :(:) :) :) :) :)” ,

Sie dürfen die Nichtterminalsymbole der am Anfang gegebenen Grammatik verwenden, ohne die Regeln dafür zu wiederholen. Mögliche Leerzeichen zwischen den einzelnen Smileys müssen Sie zur Vereinfachung hier nicht beachten.

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

LÖSUNGSVORSCHLAG:

Wir haben das 4-Tupel (Σ, V, S, P) :

$$\begin{aligned}\Sigma &= \{ \text{“:”}, \text{“,”}, \text{“8”}, \text{“=”}, \text{“o”}, \text{“-”}, \text{“)”}, \text{“\{”}, \text{“D”}, \text{“(”}, \text{“|”} \} \\ V &= \left\{ \langle \textit{SmileySequence} \rangle, \langle \textit{TraurigerSmiley} \rangle, \langle \textit{FröhlicherSmiley} \rangle, \right. \\ &\quad \left. \langle \textit{Nase} \rangle, \langle \textit{FröhlicherMund} \rangle, \langle \textit{TraurigerMund} \rangle \right\} \\ S &= \langle \textit{SmileySequence} \rangle \\ P &= \left\{ \begin{array}{l} \langle \textit{SmileySequence} \rangle ::= \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \quad | \langle \textit{TraurigerSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle \langle \textit{FröhlicherSmiley} \rangle [\langle \textit{SmileySequence} \rangle] \\ \text{sowie alle ursprünglichen Produktionen außer } \langle \textit{Smiley} \rangle \end{array} \right\}\end{aligned}$$

Es gibt zahlreiche akzeptable Lösungsmöglichkeiten, hier nur durch die Produktionen

beschrieben (Rest des 4-Tupels ändert sich entsprechend):

$$\begin{aligned}\langle \textit{SmileySequence} \rangle &::= (\langle \textit{SeqElem} \rangle)^+ \\ \langle \textit{SeqElem} \rangle &::= [\langle \textit{Smiley} \rangle (\langle \textit{FröhlicherSmiley} \rangle)^+] \langle \textit{FröhlicherSmiley} \rangle\end{aligned}$$

oder auch

$$\begin{aligned}\langle \textit{SmileySequence} \rangle &::= \langle \textit{HappySmileySeq} \rangle \mid \langle \textit{SadSmileySeq} \rangle \\ \langle \textit{HappySmileySeq} \rangle &::= \langle \textit{FröhlicherSmiley} \rangle \mid \langle \textit{FröhlicherSmiley} \rangle \langle \textit{SmileySequence} \rangle \\ \langle \textit{SadSmileySeq} \rangle &::= \langle \textit{SadSmileyPart} \rangle \mid \langle \textit{SadSmileyPart} \rangle \langle \textit{SmileySequence} \rangle \\ \langle \textit{SadSmileyPart} \rangle &::= \langle \textit{TraurigerSmiley} \rangle \langle \textit{HappySmileySeq} \rangle \mid \langle \textit{HappySmileySeq} \rangle\end{aligned}$$

Wichtig ist, dass die Grammatik keine leere Sequenz erlaubt, sowohl fröhlich oder traurig beginnen kann und auf jedem traurigen Smiley mindestens 2, also insbesondere auch mehr als 2 fröhliche Smileys folgen können, und dass mehr als ein trauriger Smiley erlaubt ist.

H5-2 Mit Schleifen malen (5 Punkte; Datei: Schleife.java)

Wir verwenden erneut die bereitgestellte Klasse `GraphicsWindow` aus Aufgabe A5-2.

Schreiben Sie ein Programm, das ein `GraphicsWindow` mit genau 640x480 Pixeln öffnet und dieses wie folgt einfärbt: Der Punkt mit den Koordinaten (x, y) soll mit dem RGB-Farbwert $(r(x, y), g(x, y), b(x, y))$ eingefärbt werden, wobei die drei Funktionswerte in ganze Zahlen im Bereich 0–255 zu konvertieren sind. Die Funktionen sind dabei wie folgt spezifiziert:

$$\begin{aligned}f(x, y) &= \left(\frac{300 - y}{64} - \sqrt{\frac{|x - 400|}{64}} \right)^2 + \left(\frac{x - 400}{64} \right)^2 \\ r(x, y) &= \begin{cases} 255 - 42 \cdot f(x, y) & \text{falls } f(x, y) \leq 4.0 \\ 255 - \sqrt{22x} + \sqrt{32y} & \text{sonst} \end{cases} \\ g(x, y) &= \begin{cases} \sqrt{100 \cdot b(x, y)} & \text{falls } f(x, y) > 4.0 \\ \frac{y}{6} & \text{sonst} \end{cases} \\ b(x, y) &= \begin{cases} \sqrt{100 \cdot r(x, y)} & \text{falls } f(x, y) > 4.0 \\ \frac{y}{3} & \text{sonst} \end{cases}\end{aligned}$$

Alle benötigten Java-Funktionen finden Sie in der Klasse `Math` aus der Standardbibliothek. Versuchen Sie Ihr Programm möglichst einfach zu gestalten.

Wenn Sie möchten, dürfen Sie dazu auch eigene statische Methoden deklarieren und verwenden; dies hat aber keinen Einfluss auf die Bewertung.

LÖSUNGSVORSCHLAG:

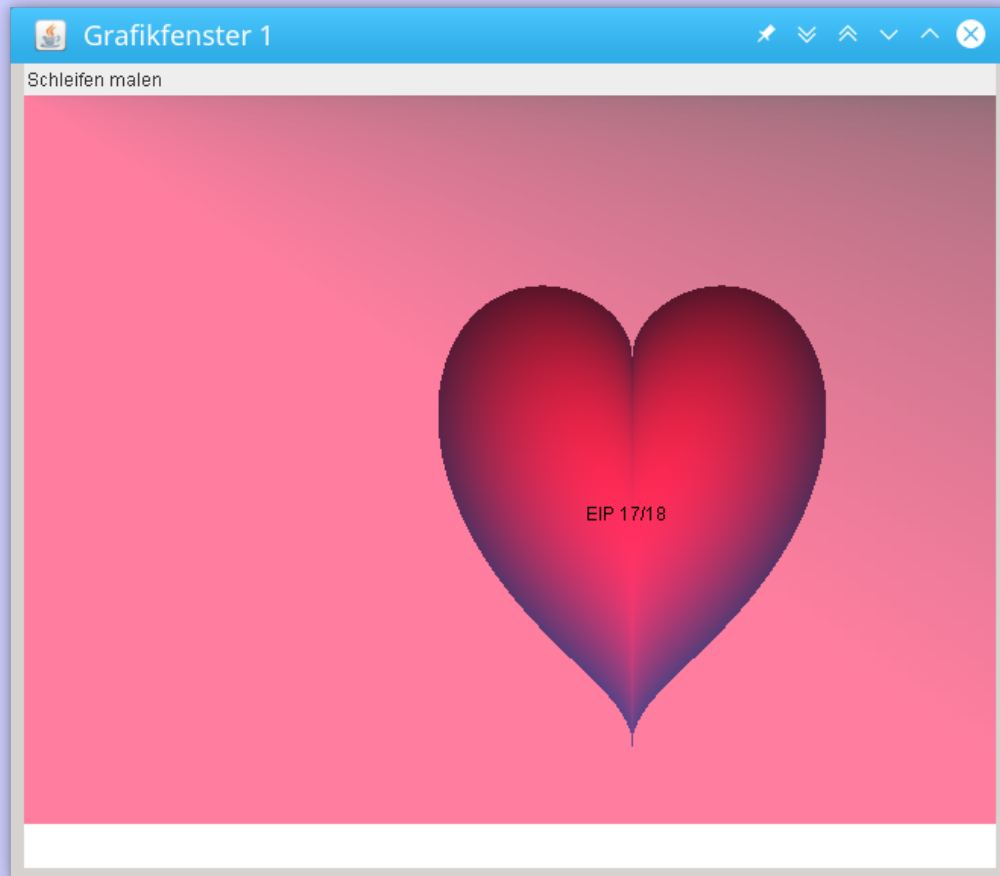
```
import java.awt.*;

public class Heart {
    final static int    MAX_X = 640; // Konstanten optional
    final static int    MAX_Y = 480;
    final static int    OFF_X = 400;
    final static int    OFF_Y = 300;
    final static double SCALE = 64;

    public static void main(String[] args) {
        GraphicsWindow window = new GraphicsWindow(MAX_X,MAX_Y);
        window.setText("Mit Schleifen malen");
        double maxf = 110.0;
        for (int px = 0; px<MAX_X; px++) {
            for (int py = 0; py<MAX_Y; py++) {
                double fval = f(px,py);
                int r = 255;
                int g;
                int b;
                if (fval <= 4.0) {
                    r = r - (int) (42*fval);
                    g = py/6;
                    b = py/3;
                } else {
                    r = r - (int) (Math.sqrt(22*px)-Math.sqrt(32*py)); // r-(x-y)=r-x+y
                    r = Math.max(0,Math.min(255,r)); // Wert beschränken!
                    b = (int) Math.sqrt(r*100);      // Reihenfolge beachten!
                    g = (int) Math.sqrt(b*100);
                }
                window.setColor(new Color(r,g,b));
                window.drawPoint(new Point(px, py));
            }
        }
        window.mouseClick();
        System.exit(0);
    }

    public static double f(double x, double y){
        x = (x-OFF_X)/SCALE;
        y = (OFF_Y-y)/SCALE;
        double res = Math.pow(y - Math.sqrt(Math.abs(x)),2);
        return res + x*x; // ...oder + Math.pow(x,2)
    } }
```

Es sollte folgendes Bild entstehen:



Abgabe: Lösungen zu den Hausaufgaben können bis Sonntag, den 26.11.17, mit UniWorX nur als **.zip** abgegeben werden. Aufgrund des Klausurbonus müssen die Hausaufgaben von Ihnen alleine gelöst werden. Abschreiben bei den Hausaufgaben gilt als Betrug und kann zum Ausschluss von der Klausur zur Vorlesung führen. Bitte beachten Sie auch die Hinweise zum Übungsbetrieb auf der Vorlesungshomepage (www.tcs.ifi.lmu.de/lehre/ws-2017-18/eip/).