Aufgabenblatt 1

Kompetenzstufe 1

Allgemeine Informationen zum Aufgabenblatt:

- Die Abgabe erfolgt in TUWEL. Bitte laden Sie Ihr IntelliJ-Projekt bis spätestens **Mittwoch, 15.11.2023 23:55 Uhr** in TUWEL hoch.
- Zusätzlich müssen Sie in TUWEL ankreuzen, welche Aufgaben Sie gelöst haben.
- Ihre Programme müssen kompilierbar und ausführbar sein.
- Ändern Sie bitte **nicht** die **Dateinamen** und die **vorhandene Ordnerstruktur**.
- Verwenden Sie, falls nicht anders angegeben, für alle Ausgaben `System.out.println()` bzw. `System.out.print()`.
- Verwenden Sie für die Lösung der Aufgaben keine Aufrufe (Klassen) aus der Java-API, außer diese sind ausdrücklich erlaubt.
- Erlaubt sind die Klassen `CodeDraw`, `Math` und `String` oder Klassen, die in den Hinweisen zu den einzelnen Aufgaben aufscheinen.

In diesem Aufgabenblatt werden folgende Themen behandelt:

- Verzweigungen (if-Anweisung, switch-Anweisung)
- Einfache Schleifen (for-Schleife, while-Schleife)
- Umgang mit den Klassen `String` und `CodeDraw`

Aufgabe 1 (1 Punkt)

Erweitern Sie die Methode `main`:

a) Schreiben Sie eine for-Schleife, die alle durch 11 teilbaren Zahlen im Intervall[1]]11, 110] aufsummiert und das Ergebnis auf der Konsole ausgibt.
Erwartetes Ergebnis: 594

b) Schreiben Sie eine for-Schleife, die jede 7. Zahl im Intervall [7, 49[(beginnend mit 7) nebeneinander durch Leerzeichen getrennt ausgibt.
Erwartetes Ergebnis: 7 14 21 28 35 42

c) Schreiben Sie eine for-Schleife, die alle Zahlen, die durch 13 und 17 im Intervall von]221, 1105] teilbar sind, hintereinander und getrennt durch Sterne (`'*'`) ausgibt. Zusätzlich wird noch vor der ersten Zahl und nach der letzten Zahl ein Stern ausgegeben.
Erwartetes Ergebnis: *442*663*884*1105*

d) Schreiben Sie eine for-Schleife, die alle Zeichen der ASCII[2]-Werte im Intervall]70, 80[in absteigender Reihenfolge durch Beistriche getrennt ausgibt.
Erwartetes Ergebnis: O,N,M,L,K,J,I,H,G

e) Schreiben Sie eine for-Schleife, die alle Vorkommen der Buchstaben `'z'` und `'Z'` im Satz `Zehn zahme Ziegen zogen zehn Zentner zerbröselten Zucker zum Zoo!` zählt.
Erwartetes Ergebnis: 10

[1] Intervall-Schreibweise: `https://de.wikipedia.org/wiki/Intervall_(Mathematik)`
[2] ASCII-Code: `https://de.wikipedia.org/wiki/American_Standard_Code_for_Information_Interchange`

Aufgabe 2 (1 Punkt)

Erweitern Sie die Methode `main`:

- Deklarieren Sie eine String-Variable `text` und initialisieren Sie diese mit `"Eine nennenswerte und geeignete Sprache."`. Testen Sie zusätzlich mit dem String `"Anzahl der Zeichen ist nicht genug!"`, um Ihre Implementierung zu prüfen. Sie dürfen auch mehrere String-Variablen deklarieren und durch entsprechende Zuweisungen unterschiedliche Tests realisieren.

a) Schreiben Sie eine while-Schleife, die vom String `text` von vorne beginnend jedes zweite Zeichen überprüft und nebeneinander ausgibt, falls es sich nicht um das Zeichen `'n'` handelt. Das erste ausgegebene Zeichen für den String `"Eine nennenswerte und geeignete Sprache."` ist in diesem Fall das Zeichen `'i'`, gefolgt von einem `'e'`. Danach kommt zweimal ein `'n'`, das jeweils ausgelassen wird und es geht dann wieder weiter mit einem `'e'`.
Erwartetes Ergebnis:
`"Eine nennenswerte und geeignete Sprache."` **liefert** `"ieeset  eit pah."`
`"Anzahl der Zeichen ist nicht genug!"` **liefert** `"aldrZihitctgg"`

b) Schreiben Sie eine while-Schleife, die den String `text` verkehrt (von hinten beginnend) durchläuft und Zeichen für Zeichen analysiert und alle Zeichen in der ursprünglichen Reihenfolge in einen neuen String schreibt, bis der Buchstabe `'n'` 5-Mal gefunden wurde. Gab es weniger als 5 Vorkommen von `'n'`, dann wird der komplette Inhalt des Strings `text` in den neuen String geschrieben. Danach wird der neue String ausgegeben.
Erwartetes Ergebnis:
`"Eine nennenswerte und geeignete Sprache."`
liefert `"nnenswerte und geeignete Sprache."`
`"Anzahl der Zeichen ist nicht genug!"`
liefert `"Anzahl der Zeichen ist nicht genug!"`

c) Schreiben Sie eine while-Schleife, die im String `text` die Vorkommen aller Buchstaben im Alphabet zwischen `'a'` bis `'m'` (beides inklusive und auch als Großbuchstaben) zählt und das Ergebnis auf der Konsole ausgibt.
Erwartetes Ergebnis:
`"Eine nennenswerte und geeignete Sprache."` **liefert** 19
`"Anzahl der Zeichen ist nicht genug!"` **liefert** 18

d) Schreiben Sie eine while-Schleife, die alle geraden durch 19 teilbaren Zahlen im Intervall]19, 304[nebeneinander durch Leerzeichen getrennt ausgibt.
Erwartetes Ergebnis: 38 76 114 152 190 228 266

Aufgabe 3 (1 Punkt)

Erweitern Sie die Methode `main`:

Implementieren Sie ein Programm, welches verschiedene Formen in unterschiedlicher Größe auf der Konsole ausgeben kann. Es gibt für die Steuerung des Programms drei Variablen. Die Art der Form wird mit den Variablen `boolean isTrapezoid` und `boolean isParallelogram` ausgewählt. Bei `boolean isTrapezoid` gleich `true` wird ein Trapez (siehe Tabelle 1) gezeichnet und bei `boolean isParallelogram` gleich `true` wird ein Parallelogramm (siehe Tabelle 2) ausgegeben. Sind beide Variablen `true`, dann werden beide Formen untereinander ausgegeben.
Mit der Variablen `int width` kann die Breite einer Seite der Form angegeben werden. Die Variable darf nur positive Werte größer 1 annehmen. Beim Trapez gibt `int width` die Breite der unteren Seitenkante an. Die Höhe ergibt sich aus der ganzzahligen Hälfte der Breite (z.B. bei Breite 8 und 9 ist die Höhe 4, bei Breite 10 und 11 ist die Höhe 5, usw.). Beim Parallelogramm gibt `int width` die obere und untere Seitenbreite an. Die Höhe ist in diesem Fall gleich der Breite.
Es reicht bei dieser Aufgabe aus, wenn Sie den Variablen konkrete Werte direkt zuweisen.

Beispiele für unterschiedliche Trapeze:

Eingabe	Konsolenausgabe	Eingabe	Konsolenausgabe

`isTr... = true;`
`isPa... = false;`
`width = 2;`
	`##`	`isTr... = true;`
		`isPa... = false;`
		`width = 3;`
			`###`

`isTr... = true;`
`isPa... = false;`
`width = 4;`
	` ##`
	`####`
		`isTr... = true;`
		`isPa... = false;`
		`width = 5;`
			` ###`
			`#####`

`isTr... = true;`
`isPa... = false;`
`width = 6;`
	`  ##`
	` ####`
	`######`
		`isTr... = true;`
		`isPa... = false;`
		`width = 7;`
			`  ###`
			` #####`
			`#######`

`isTr... = true;`
`isPa... = false;`
`width = 10;`
	`    ##`
	`  ####`
	` ######`
	`########`
	`##########`
		`isTr... = true;`
		`isPa... = false;`
		`width = 11;`
			`    ###`
			`  #####`
			` #######`
			`#########`
			`###########`

Tabelle 1: Ausgaben für Trapeze (von links oben nach rechts unten) mit `width` 4, 5, 6, 7, 10 und 11.

Beispiele für unterschiedliche Parallelogramme:

Eingabe	Konsolenausgabe	Eingabe	Konsolenausgabe

`isTr... = false;`
`isPa... = true;`
`width = 2;`
	`##`
	` ##`
		`isTr... = false;`
		`isPa.. = true;`
		`width = 3;`
			`###`
			` # #`
			`  ###`

`isTr... = false;`
`isPa... = true;`
`width = 4;`
	`####`
	` #  #`
	`  #  #`
	`   ####`
		`isTr... = false;`
		`isPa.. = true;`
		`width = 5;`
			`#####`
			` #   #`
			`  #   #`
			`   #   #`
			`    #####`

`isTr... = false;`
`isPa... = true;`
`width = 6;`
	`######`
	` #    #`
	`  #    #`
	`   #    #`
	`    ######`
		`isTr... = false;`
		`isPa... = true;`
		`width = 7;`
			`#######`
			` #     #`
			`  #     #`
			`   #     #`
			`    #     #`
			`     #     #`
			`      #######`

`isTr... = false;`
`isPa... = true;`
`width = 10;`
	`##########`
	` #        #`
	`  #        #`
	`   #        #`
	`    #        #`
	`     #        #`
	`      #        #`
	`       #        #`
	`        #        #`
	`         ##########`
		`isTr... = false;`
		`isPa... = true;`
		`width = 11;`
			`###########`
			` #         #`
			`  #         #`
			`   #         #`
			`    #         #`
			`     #         #`
			`      #         #`
			`       #         #`
			`        #         #`
			`         #         #`
			`          ###########`

Tabelle 2: Ausgaben für Parallelogramme (von links oben nach rechts unten) mit `width` 4, 5, 6, 7, 10 und 11.

Aufgabe 4 (1 Punkt)

Erweitern Sie die Methode `main`:

- Implementieren Sie ein Programm, welches das in Abbildung 1 gezeigte Bild ausgibt.
- Im Bild sind alle Maßangaben vorhanden, die notwendig sind, um dieses Bild zu erzeugen.
- Das gesamte Bild hat eine Größe von 400×400 Pixel und der Punkt $P(x = 0, y = 0)$ befindet sich in der oberen linken Ecke. Dazu erstellen Sie ein Objekt der Klasse `CodeDraw` mit der Anweisung `CodeDraw myDrawObj = new CodeDraw(400, 400)`, um ein Zeichenfenster mit der Größe 400×400 Pixel zu erstellen.
- Mit `myDrawObj.setLineWidth(2)` wird die Linienbreite auf 2 Pixel gesetzt.
- Mit `myDrawObj.setColor(Palette.RED)` können Sie die aktuelle Zeichenfarbe auf rot setzen. Andere Farben können entsprechend gesetzt werden. Verwendete Farben in der Abbildung: `MAGENTA`, `GRAY`, `BLUE`, `YELLOW`, `RED`, `ORANGE` und `GREEN`.
- Verwenden Sie für die grünen und orangen Linien **eine** einzige Schleife. Reihenfolge kann beliebig gewählt werden.
- Um Zeichnungen im Ausgabefenster sichtbar zu machen, müssen Sie die Methode `myDrawObj.show()` aufrufen.
- Die Dokumentation unter dem Link `https://krassnig.github.io/CodeDrawJavaDoc/v4.0.x/codedraw/package-summary.html` kann Ihnen dabei helfen, diese Aufgabe zu lösen.

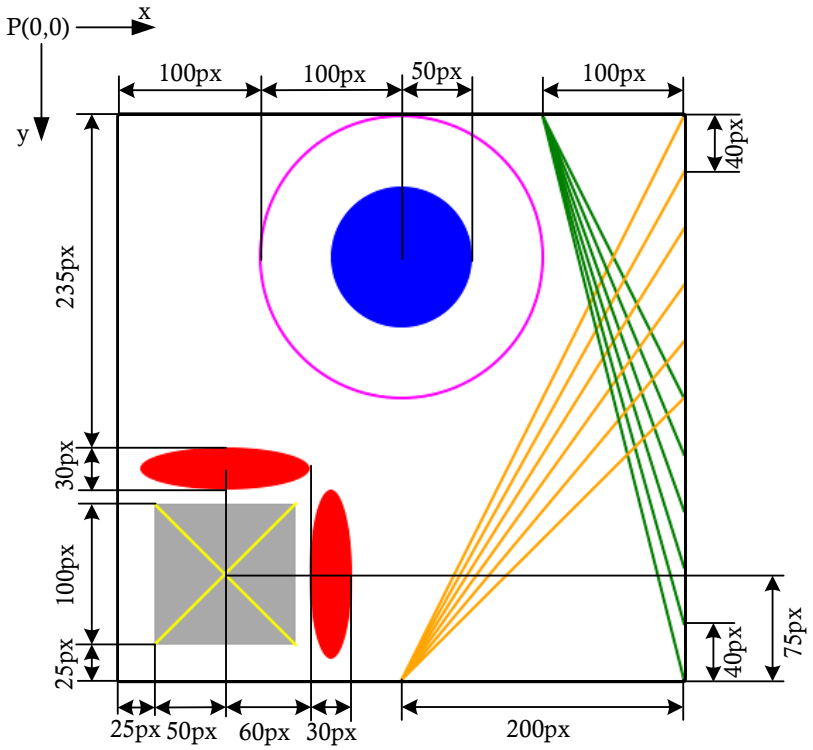


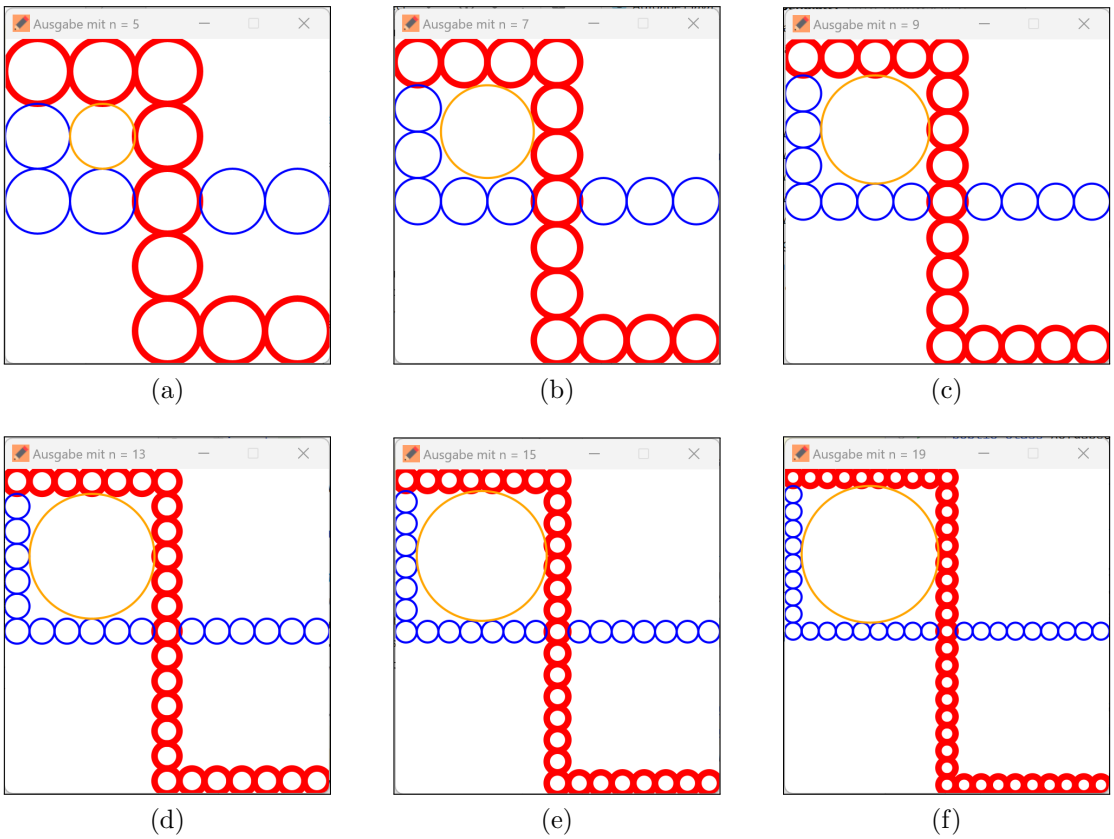
Abbildung 1: Ergebnisbild mit den entsprechenden Maßangaben.

Aufgabe 5 (2 Punkte)

Erweitern Sie die Methode `main`:

- Erstellen Sie ein Ausgabefenster mit der Größe 300×300 Pixel.
- Schreiben Sie ein Programm, das unter Vorgabe einer Variable `n` die Abbildungen 2a-2f erzeugen kann. Die Variable `n` gibt an, wie viele Kreise maximal horizontal und vertikal im CodeDraw-Fenster Platz haben sollen und sie darf dabei nur ungerade Werte größer gleich 5 und kleiner gleich 19 annehmen. Überprüfen Sie in Ihrer Implementierung, ob die Vorgaben für `n` eingehalten werden, ansonsten geben Sie einen Hinweis (Fehlermeldung) auf der Konsole aus. Der orange Kreis und die blauen Kreise haben eine Liniendicke von 2 Pixel. Die roten Kreise haben eine Liniendicke von 6 Pixel. Geben Sie acht, dass die roten vor den blauen Kreisen gezeichnet werden und die blauen Kreise bei den Kreuzungspunkten die roten Kreise nicht überzeichnen, d.h. kein blauer Kreis sollte vollständig über einem roten Kreis gezeichnet werden. Die partielle Überzeichnung durch die Kreise vor und nach den Kreuzungspunkten ist erlaubt.

(!) Hinweis: Verwenden Sie Gleitkommaarithmetik für die Berechnung des Radius, um für jedes `n` das CodeDraw-Fenster füllend auszunutzen. Die Variable `n` wird direkt bei der Initialisierung angegeben und muss nicht über einen User-Input realisiert und eingegeben werden.



(a) (b) (c)

(d) (e) (f)

Abbildung 2: Verschiedene Kreismuster mit unterschiedlicher Anzahl an Kreisen `n`. a) `n = 5`, b) `n = 7`, c) `n = 9`, d) `n = 13`, e) `n = 15` und f) `n = 19`.