Aufgabenblatt 1

Kompetenzstufe 1

Allgemeine Informationen zum Aufgabenblatt:

- Die Abgabe erfolgt in TUWEL. Bitte laden Sie Ihr IntelliJ-Projekt bis spätestens **Donnerstag, 04.11.2021 20:00 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) Implementieren Sie eine Verzweigungsstruktur, die bei einem gegebenen String alle Zeichen vor dem ersten Punkt des Strings entfernt. Diese entfernten Zeichen ersetzen alle Zeichen nach dem letzten Punkt des gegebenen Strings. Die Punkte selbst bleiben an den gefundenen Stellen erhalten. Danach wird der veränderte String auf der Konsole ausgegeben. Sind nur ein oder kein Punkt vorhanden, dann wird der originale String ausgegeben. Wenn mehr als zwei Punkte vorhanden sind, dann werden alle Punkte zwischen dem ersten und letzten Punkt ignoriert. Für die Implementierung dürfen keine Schleifen verwendet werden.

Beispiele:

`".Der erste Satz. Der zweite Teil!"` liefert `".Der erste Satz."`
`"Zwei. Punkte im Satz."` liefert `". Punkte im Satz.Zwei"`
`"Mehr. als. zwei. Punkte gefunden!"` liefert `". als. zwei.Mehr"`
`"Ein Punkt vorhanden."` liefert `"Ein Punkt vorhanden."`
`"Kein Punkt vorhanden!"` liefert `"Kein Punkt vorhanden!"`

b) Implementieren Sie eine Verzweigungsstruktur, die bei einem gegebenen String überprüft, ob ein gegebenes Zeichen `character` vorkommt. Wenn das Zeichen `character` nicht im gegebenen String vorkommt, dann wird der String `"Kein Zeichen gefunden"` ausgegeben. Wenn das Zeichen `character` einmal im gegebenen String vorkommt, dann wird `"Genau ein Zeichen gefunden"` ausgegeben und wenn `character` mehr als einmal vorkommt, dann wird der String `"Zwei oder mehr Zeichen gefunden"` ausgegeben. Für die Implementierung dürfen keine Schleifen verwendet werden.

Beispiele:

`"Flussschifffahrt"` und `character = 'm'` liefert `"Kein Zeichen gefunden"`
`"Flussschifffahrt"` und `character = 'i'` liefert `"Genau ein Zeichen gefunden"`
`"Flussschifffahrt"` und `character = 'h'` liefert `"Zwei oder mehr Zeichen gefunden"`
`"Flussschifffahrt"` und `character = 's'` liefert `"Zwei oder mehr Zeichen gefunden"`

c) Implementieren Sie einen Kalenderrechner, der bei einer Angabe von Monat `int month` und Jahr `int year` die korrekte Anzahl an Tagen für diesen Monat ausgibt. Für den Monat Februar müssen Sie die Schaltjahre berücksichtigen. Beim *Gregorianischen Kalender* kommen die folgenden drei Kriterien zur Anwendung, nach denen geprüft wird, ob es sich um ein Schaltjahr handelt:

1. Schaltjahre müssen durch 4 teilbar sein.
2. Ist das Jahr auch durch 100 teilbar, ist es kein Schaltjahr, es sei denn...
3. ...das Jahr ist ebenfalls durch 400 teilbar – dann ist es ein Schaltjahr.

Beispiele: Die Jahre 2000 und 2400 sind Schaltjahre (infolge Regel 3); Jahre 1800, 2100, 2200 und 2500 sind hingegen keine Schaltjahre (infolge Regel 2, aber Regel 3 greift nicht).

Verwenden Sie für die Implementierung eine switch-Anweisung. Bei einer case-Marke dürfen if-Anweisungen verwendet werden. Bei einem falschen Datum soll der String `"Ungültiges Datum!"` ausgegeben werden, ansonsten die Anzahl der Tage des angegebenen Monats im angegebenen Jahr.

Aufgabe 2 (1 Punkt)

Erweitern Sie die Methode `main`:

a) Schreiben Sie eine for-Schleife, die alle durch 9 teilbaren Zahlen im Intervall [9, 99] aufsummiert und das Ergebnis auf der Konsole ausgibt.
Erwartetes Ergebnis: 594

b) Schreiben Sie eine for-Schleife, die jede 8. Zahl im Intervall]60, 100] (beginnend mit 61) nebeneinander mit einem Leerzeichen getrennt ausgibt.
Erwartetes Ergebnis: 61 69 77 85 93

c) Schreiben Sie eine for-Schleife, die alle durch 5 und 9 teilbaren Zahlen im Intervall von]45, 450[in einem String hintereinander und getrennt durch Punkte speichert und danach ausgibt.
Erwartetes Ergebnis: .90.135.180.225.270.315.360.405.

d) Schreiben Sie eine for-Schleife, die alle Buchstaben der ASCII [1]-Werte im Intervall [105, 114[in absteigender Reihenfolge ausgibt.
Erwartetes Ergebnis: q p o n m l k j i

e) Schreiben Sie eine for-Schleife, die alle Vorkommen des Buchstabens `'e'` im Satz `she sells sea shells on the sea shore` zählt.
Erwartetes Ergebnis: 7

[1] ASCII-Code: `https://de.wikipedia.org/wiki/American_Standard_Code_for_Information_Interchange`

Aufgabe 3 (1 Punkt)

Erweitern Sie die Methode `main`:

- Deklarieren Sie eine String-Variable `text` und initialisieren Sie diese mit `"Die Sonne scheint in vielen Ländern."`. Testen Sie zusätzlich mit dem String `"Sommer, Sonne, Kaktus"`, um Ihre Implementierung zu prüfen.

a) Schreiben Sie eine while-Schleife, die den String `text` von vorne beginnend durchläuft und Zeichen für Zeichen nebeneinander ausgibt, bis der Buchstabe `'n'` 3-Mal gefunden wurde. Gab es weniger als 3 Vorkommen von `'n'`, dann wird der komplette String ausgegeben.
Erwartetes Ergebnis:
`"Die Sonne scheint in vielen Ländern."` **liefert** `"Die Sonne schein"`
`"Sommer, Sonne, Kaktus"` **liefert** `"Sommer, Sonne, Kaktus"`

b) Schreiben Sie eine while-Schleife, die im String `text` von vorne beginnend das erste Vorkommen des Buchstabens `'l'` sucht. Wird der Buchstabe `'l'` gefunden, dann wird dessen Index auf der Konsole ausgegeben. Andernfalls wird -1 ausgegeben. Für diese Implementierung dürfen die Methoden `indexOf(...)` und `lastIndexOf(...)` **nicht** verwendet werden.
Erwartetes Ergebnis:
`"Die Sonne scheint in vielen Ländern."` **liefert** 24
`"Sommer, Sonne, Kaktus"` **liefert** -1

c) Schreiben Sie eine while-Schleife, die alle geraden durch 21 teilbaren Zahlen im Intervall]26, 280[nebeneinander ausgibt.
Erwartetes Ergebnis: 42 84 126 168 210 252

d) Schreiben Sie eine while-Schleife, die vom String `text` von hinten beginnend jedes zweite Zeichen überprüft und nebeneinander ausgibt, falls es sich nicht um das Zeichen `'s'` oder `'S'` handelt. Das erste ausgegebene Zeichen für den String `"Die Sonne scheint in vielen Ländern."` ist in diesem Fall das Zeichen `'n'`, dann `'e'`, usw.
Erwartetes Ergebnis:
`"Die Sonne scheint in vielen Ländern."` **liefert** `"nenLnli itiheneD"`
`"Sommer, Sonne, Kaktus"` **liefert** `"ukK,no rmo"`

e) Schreiben Sie eine while-Schleife, die im String `text` die Vorkommen aller Leerzeichen und die Satzzeichen `'.'` und `','` zählt und das Ergebnis auf der Konsole ausgibt.
Erwartetes Ergebnis:
`"Die Sonne scheint in vielen Ländern."` **liefert** 6
`"Sommer, Sonne, Kaktus"` **liefert** 4

Aufgabe 4 (1 Punkt)

Erweitern Sie die Methode `main` um folgende Funktionalität:

- 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`, `CYAN`, `BLUE`, `YELLOW`, `RED`, `LIME` und `ORANGE`.
- Verwenden Sie für die magentafarbenen und cyanfarbenen Linien eine einzige Schleife. Beginnend links unten nach rechts laufend soll jede dritte Linie mit der Farbe Cyan eingefärbt 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/codedraw/CodeDraw.html` kann Ihnen dabei helfen, diese Aufgabe zu lösen.

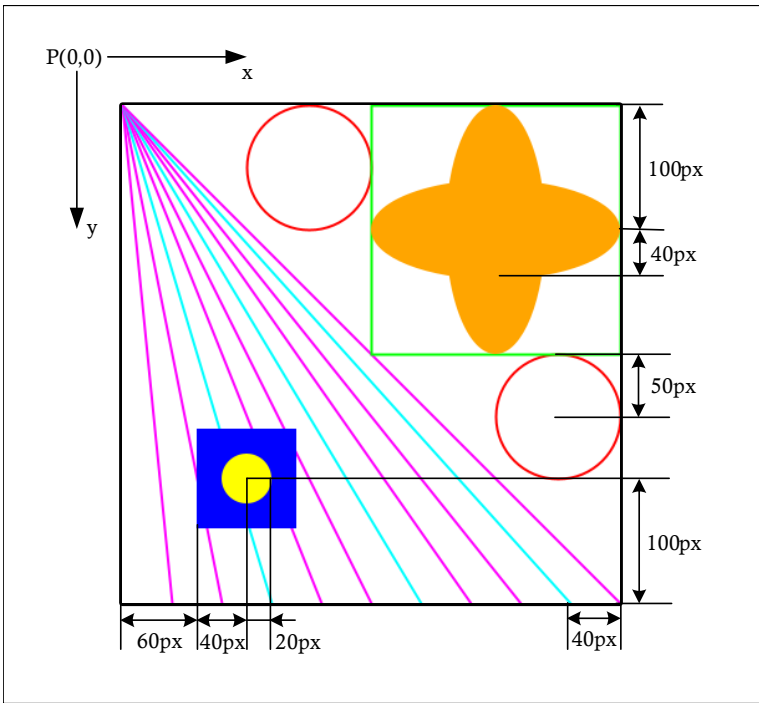


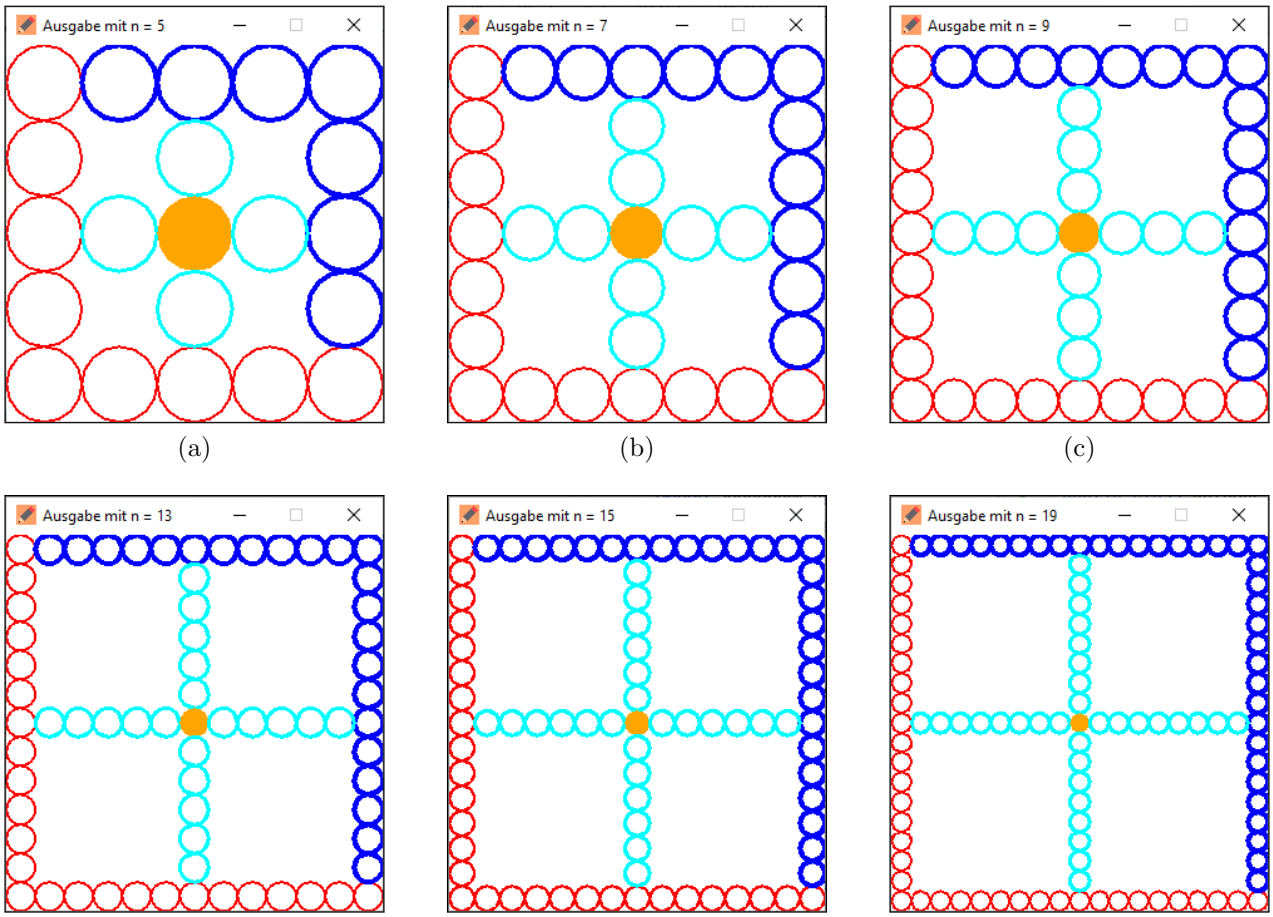
Abbildung 1: Ergebnisbild mit den entsprechenden Maßangaben

Aufgabe 5 (1 Punkt)

Erweitern Sie die Methode `main`:

- Erstellen Sie ein Ausgabefenster mit der Größe 300×300 Pixel.
- Schreiben Sie ein Programm, das unter Vorgabe einer Variablen `n` die Abbildungen 2a-2f erzeugen kann. Die Variable `n` gibt an, wie viele Kreise horizontal und vertikal zu zeichnen sind und darf dabei nur ungerade Werte größer gleich 5 annehmen.
- Die roten Kreise haben eine Liniendicke von 2 Pixel und die blauen Kreise eine Liniendicke von 4 Pixel. Die Kreise mit der Farbe Cyan haben die Liniendicke von 3 Pixel. Zusätzlich wird in der Mitte des Bildes ein gefüllter Kreis mit der Farbe Orange gezeichnet.

(!) Hinweis: Verwenden Sie Gleitkommaarithmetik, um für jedes `n` das CodeDraw-Fenster füllend auszunutzen.



(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`.