Übungen zu Algorithmen und Datenstrukturen für Medizintechnik
Wintersemester 2021/2022
Jonas Auernheimer, Alina Schüller

Lehrstuhl für sensorische Neurotechnologie
(https://www.neurotech.tf.fau.eu)

Aufgabenblatt 1 vom 25.10.2021, Abgabe am 07.11.2021

»Eine neue Programmiersprache lernt man nur, wenn man in ihr Programme schreibt.«
– *Dennis M. Ritchie, Entwickler von Unix und C*

Es reicht daher nicht aus, die Tafel- bzw. Rechnerübung zu besuchen, Sie müssen auch eigenständig zu Hause programmieren üben!

Aufgabe 1.1: Hello World!

3 Punkte
IntelliJ, Ausgabe

»Hello World!« ist ein sehr einfaches Computer-Programm, das den Satz »Hello World!« auf dem Bildschirm ausgibt. Es wird gewöhnlich dazu verwendet, die Grundlagen der Syntax einer Programmiersprache zu verdeutlichen.

Ihre Aufgabe ist es, das »Hello World!«-Programm in Java zu schreiben, zu übersetzen und auszuführen.

1. Starten Sie IntelliJ IDEA und legen Sie ein neues Java-Projekt `01-HelloWorld` an.
2. Erstellen Sie eine neue Klasse `HelloWorld` mit einer `main`-Methode.
3. Geben Sie auf dem `stdout`-Kanal den Text

   ```
   Hello World!
   ```

   gefolgt von einem *Zeilenumbruch* aus.
4. Führen Sie das Programm aus. Erscheint im Konsolen-Fenster der oben genannte Text, dann haben Sie alles richtig gemacht!
5. Noch schöner wäre es natürlich, wenn das Programm seine Schöpfer auch persönlich begrüßen würde. Legen Sie hierfür zwei Variablen geeigneten Datentypes an, in denen Sie die Namen von beiden Übungspartnern speichern. Sollten Sie noch keinen Übungspartner haben, legen Sie trotzdem beide Variablen an und denken Sie sich einen Wert für die zweite aus.

   Haben Sie noch keinen Übungspartner? Schauen Sie doch mal ins AuD-MT-Forum im StudOn!
6. Geben Sie folgende Zeile auf `stdout` aus:

   `Hello` ⟨*name1*⟩ `and` ⟨*name2*⟩`!`

   Anstelle von ⟨...⟩ soll Ihr Programm den aktuellen Wert der entsprechenden Variable ausgeben.
7. Geben Sie die Datei `HelloWorld.java` ab.

   Stehen Ihre Namen als Kommentare am Anfang der Datei?

Aufgabe 1.2: Theorie: Datentypen 10 Punkte

Datentypen

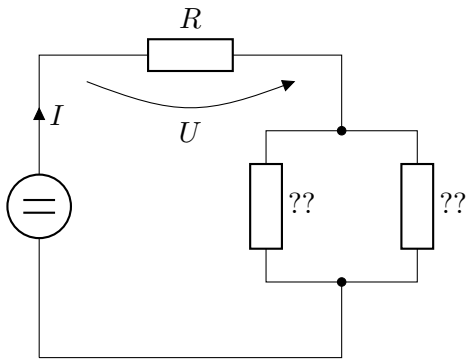
Aufgabenstellung und Abgabe als E-Test im StudOn.

Wichtig: Die Aufgaben im StudOn sind Einzelabgaben und von jedem Studierenden **individuell** zu bearbeiten. Falls nur eine Person in der Gruppe die StudOn-Aufgabe bearbeitet, bekommt die andere Person **keine Punkte** auf die entsprechende Aufgabe!

Aufgabe 1.3: Schaltungsanalyse 16 Punkte

Variablen, Operationen

In der Elektrotechnik müssen häufig Schaltungen ausgewertet werden[1]. In dieser Aufgabe sollen Sie sich mit der Analyse der nachfolgenden Schaltung auseinandersetzen. Die Schaltung setzt sich aus einer Spannungsquelle, einem Vorwiderstand und einer Steckplatte zusammen, in welche entweder zwei parallel geschaltete Plattenkondensatoren, zwei parallel geschaltete Widerstände oder zwei Kurzschlüsse (also einfache Verbindungen ohne Bauteile) gesteckt werden können.



Einige Parameter und Konstanten der Schaltung sind bereits bekannt:

Bezeichnung	**Wert**	**Variablennname in *Java***
Vorwiderstand		
R_{vor}	??? Ω	`protectiveResistor`
U	5 V	`voltage`
Plattenkondensatoren		
d_1	3 mm	`d1`
d_2	4 mm	`d2`
A_1	9.7 cm^2	`a1`
A_2	5.1 cm^2	`a2`
Widerstände		
R_1	1 $k\Omega$	`resistor1`
R_2	472 Ω	`resistor2`
ε_0	$8.854 \cdot 10^{-12}$ $\frac{As}{Vm}$	`EPSILON_0`

Tabelle 1: Schaltungsparameter und -konstanten und entsprechende Variablennamen

[1] Sie sollten diese Aufgabe auch ohne große Kenntnisse der Elektrotechnik bearbeiten können, da alle notwendigen Schritte im folgenden erklärt werden.

Bei Ihren Berechnungen sollen die Stromstärke I und die Leistung P, die am Vorwiderstand R_{vor} verbraucht wird, berechnet werden, sowie – je nach Schaltungsaufbau – die Kapazitäten der einzelnen Kondensatoren C_1 und C_2 und die Gesamtkapazität C_{ges} oder der Gesamtwiderstand R_{ges} der Parallelschaltung zwischen R_1 und R_2.

Um das Programm interaktiver zu gestalten soll der Wert des Vorwiderstandes veränderbar sein, also bei jeder Ausführung des Programmes neu eingegeben werden können. Ebenso soll die Steckplatte bei jeder Programmausführung neu konfiguriert werden können. Aber keine Angst – wir werden Ihnen anhand der folgenden Anleitung zeigen, wie Sie das Programm Stück für Stück entwickeln und neue Funktionen hinzufügen:

1. **Projekt anlegen:**
   Erstellen Sie ein neues Projekt namens `01-Circuit`, sowie die Klasse `Circuit` mit einer `main`-Methode.

2. **Variablen und Konstanten anlegen:**
   Legen Sie Variablen für die in Tabelle 1 gegebenen Parameter an und initialisieren Sie diese entsprechend. Überlegen Sie sich hierfür geeignete Datentypen.

   Hinweise: Denken Sie daran, die Schaltungsparameter zuerst in die richtigen Einheiten umzurechnen, also z. B. $cm^2 \rightarrow m^2$!
   Da es sich bei ε_0 und ε_r um Naturkonstanten handelt, sollen diese natürlich auch in *Java* als **Konstanten** definiert werden. Aber da die Variablennamen in Tabelle 1 als einzige in Großbuchstaben definiert wurden, haben Sie sich das bestimmt schon gedacht... ☺

3. **Vorwiderstand dynamisch festlegen:**
   Wie bereits erwähnt soll der Wert des Vorwiderstandes bei jeder Ausführung des Programmes vom Benutzer veränderbar sein. Dafür bedienen wir uns der bereits in der Java-Standardbibliothek vorhandenen Klasse `Scanner`, die Text von der Standardeingabe (`stdin`) lesen kann.

   Binden Sie diese andere Klasse ein, indem Sie ganz oben im Programm folgende Zeile einfügen:

   ```
   import java.util.Scanner;
   ```

   Um nun den Wert des Widerstandes festlegen zu können, müssen Sie ein Objekt `scRes` der Klasse `Scanner` erstellen. Initialisieren Sie nun die vorhin angelegte Variable `protectiveResistor` mit dem Befehl `scRes.nextDouble();`, der die nächste Eingabe auf `stdin` als *Gleitkommazahl* interpretiert. Lassen Sie davor einen sinnvollen Hinweis auf der Standardausgabe ausgeben, damit das Programm später auch von anderen Benutzern gut verwendet werden kann.

   Beachten Sie beim Testen Ihres Programmes, dass die Eingabe der Gleitkommazahl von der Sprache Ihres Betriebssystems abhängt. Wenn Ihr Betriebssystem auf Deutsch eingestellt ist, müssen Sie die Gleitkommazahl mit einem Komma eingeben, ist die Sprache auf Englisch, benutzen Sie einen Dezimalpunkt.

   Hinweis: Ein Objekt der Klasse `Scanner` mit Namen `scRes`, das von `stdin` liest, wird wie folgt erstellt:
   `Scanner scRes = new Scanner(System.in);`
   Fügen Sie diese Codezeile Ihrem Programm hinzu.

4. **Falsche Benutzereingaben behandeln:**
   In Ihrer Schaltung sollen nur **echt positive** Vorwiderstandswerte ($R_{vor} > 0$) auftreten. Stellen Sie daher sicher, dass der initialisierte Widerstandswert im Wertebereich liegt. Falls dies nicht der Fall ist, soll eine aussagekräftige Fehlermeldung auf `stderr` ausgegeben werden.

5. **Stromstärke und Leistung berechnen:**
   Berechnen Sie die Stromstärke I und speichern Sie das Ergebnis in einer Variable `current` geeigneten Datentyps. Verwenden Sie zur Berechnung der Stromstärke I das Ohmsche Gesetz in Gleichung 1. Berechnen Sie zudem die Leistung, die am Vorwiderstand R_{vor} verbraucht wird, mithilfe der Formel aus Gleichung 2. Speichern Sie das Ergebnis in einer Variable `power` geeigneten Typs.

$$I = \frac{U}{R_{vor}} \tag{1}$$

$$P = I^2 \cdot R_{vor} \tag{2}$$

   Geben Sie die berechneten Werte anschließend wie folgt auf der Standardausgabe aus:

```
Current: ⟨current⟩ Ampere; Power: ⟨power⟩ Watt
```

6. **Steckplatte konfigurieren:**
   Widmen Sie sich nun der Steckplatte. Da je nach Konfiguration parallel geschaltete Widerstände, Kondensatoren oder Kurzschlüsse (einfache Verbindungen ohne Bauteile) verbaut sind, benötigt Ihr Programm eine Funktion, in welcher der Benutzer bei Programmstart auswählen kann, wie die Schaltung aufgebaut ist. Dafür bedienen wir uns wieder der Klasse `Scanner`:

   a) Legen Sie ein neues Scanner-Objekt mit Namen `scMod` mittels `Scanner scMod = new Scanner(System.in);` an.

   b) Initialisieren Sie eine neu angelegte Variable `module` mit dem Befehl **`scMod.nextLine();`**, der die nächste Eingabe auf `stdin` als *String* interpretiert. Lassen Sie zuvor wieder einen sinnvollen Hinweis an den Benutzer auf `stdout` ausgeben!

   c) Benutzen Sie zur Unterscheidung der verschiedenen Steckplatten-Module eine `switch-case`-Kontrollstruktur:

   - `module` $\hat{=}$ `"cap"` $\rightsquigarrow$ *Kondensatoren*
   - `module` $\hat{=}$ `"res"` $\rightsquigarrow$ *Widerstände*
   - `module` $\hat{=}$ `"short"` $\rightsquigarrow$ *Kurzschlüsse*

   d) Bei allen anderen Eingaben soll eine aussagekräftige Fehlermeldung auf `stderr` ausgegeben werden.

7. **Steckplatte mit Kondensatoren:**

   a) Die Kapazität eines Plattenkondensators wird mittels folgender Formel berechnet:

$$C_i = \frac{\varepsilon_0 \cdot \varepsilon_r \cdot A_i}{d_i} \tag{3}$$

   ε_0 steht dabei für die Elektrische Feldkonstante, A_i für die Plattenfläche und d_i für den Plattenabstand. ε_r steht für die Dielektrizitätszahl und kann hier als 1 angenommen werden, da sich zwischen den beiden Kondensatorplatten Luft befindet.

   b) Berechnen Sie die Kapazitäten der Kondensatoren und speichern Sie die Werte in Variablen `capacity1` und `capacity2` geeigneten Datentyps. Geben Sie die Werte der Variablen wie folgt auf der Standardausgabe aus:

```
Capacity1: ⟨capacity1⟩ Farad
Capacity2: ⟨capacity2⟩ Farad
```

c) Nun soll noch die Gesamtkapazität der Schaltung berechnet werden. Für die Parallelschaltung von n Kapazitäten gilt folgende Formel:

$$C_{ges} = \sum_{i=1}^{n} C_i \tag{4}$$

d) Speichern Sie das Ergebnis in eine Variable `totalCapacity` und geben Sie den Wert wie folgt auf `stdout` aus:

```
The total capacity is: ⟨totalCapacity⟩ Farad.
```

8. **Steckplatte mit Widerständen**:

a) Berechnen Sie den Gesamtwiderstand der beiden parallel geschalteten Widerstände. Für die Parallelschaltung von n Widerständen gilt folgende Formel:

$$\frac{1}{R_{ges}} = \sum_{i=1}^{n} \frac{1}{R_i} \tag{5}$$

b) Speichern Sie das Ergebnis in eine Variable `totalResistance` und geben Sie den Wert wie folgt auf `stdout` aus:

```
The total resistance is: ⟨totalResistance⟩ Ohm.
```

9. **Steckplatte mit Kurzschlüssen**:

a) Geben Sie lediglich folgenden Text auf `stdout` aus:

```
Board shorted - No modules installed!
```

Hinweis: Schließen Sie die beiden `Scanner`-Objekte am Ende der `main`-Methode wie folgt:

```
scRes.close();
scMod.close();
```

10. **Programm auf Randbedingungen und mögliche Fehler überprüfen**:
Überprüfen Sie, ob Ihr Programm einwandfrei funktioniert, testen Sie es mit verschiedenen Widerstandswerten und geben Sie dann die Datei `Circuit.java` ab.

Stehen Namen und Matrikelnummern am Anfang der Datei?

Sollte Ihr Programm nicht übersetz- bzw. ausführbar sein, wird die Lösung mit 0 Punkten bewertet. Stellen Sie also sicher, dass IntelliJ IDEA keine Fehler in Ihrem Programm anzeigt, Ihr Programm übersetz- und ausführbar ist sowie die in der Aufgabenstellung vorgegebenen Namen und Schnittstellen *exakt* eingehalten werden.