Übungsblatt 3 (Abgabe bis Dienstag, den 5.11.2024, um 12:00 Uhr)

Prof. Dr. J. Giesl
D. Cloerkes, J.-C. Kassing, N. Lommen, E. Meyer

Allgemeine Hinweise:

- Die **Deadline** zur **Abgabe** der Hausaufgaben ist am **Dienstag, den 5.11.2024, um 12:00 Uhr**.
- Der **Workflow** sieht wie folgt aus. Die Abgabe der Hausaufgaben erfolgt **im Moodle-Lernraum** und kann nur in **Zweiergruppen** stattfinden. Dabei müssen die Abgabepartner*innen **dasselbe Tutorium** besuchen. Nutzen Sie ggf. das entsprechende **Forum** im Moodle-Lernraum, um eine*n Abgabepartner*in zu finden. Es darf **nur ein*e** Abgabepartner*in die Abgabe hochladen. Diese*r muss sowohl die **Lösung** als auch den **Quellcode** der Programmieraufgaben hochladen. Die Bepunktung wird dann von uns für **beide** Abgabepartner*innen **separat** im Lernraum eingetragen. Die Feedbackdatei ist jedoch nur dort sichtbar, wo die Abgabe hochgeladen wurde und muss innerhalb des Abgabepaars **weitergeleitet** werden.
- Die **Lösung** muss als PDF-Datei hochgeladen werden. Damit die Punkte beiden Abgabepartner*innen zugeordnet werden können, müssen **oben** auf der **ersten Seite** Ihrer Lösung die **Namen**, die **Matrikelnummern** sowie die **Nummer des Tutoriums** von **beiden** Abgabepartner*innen angegeben sein.
- Der **Quellcode** der Programmieraufgaben muss als `.zip`-Datei hochgeladen werden und **zusätzlich** in der PDF-Datei mit Ihrer Lösung enthalten sein, sodass unsere Hiwis ihn mit Feedback versehen können.

  Auf diesem Blatt muss Ihre Codeabgabe Ihren vollständigen Java-Code in Form von `.java`-Dateien enthalten. Aus dem Lernraum heruntergeladene Klassen, etwa die Datei `SimpleIO.java`, dürfen nicht mit abgegeben werden.

  Stellen Sie sicher, dass Ihr Programm von `javac` **akzeptiert** wird, wenn die entsprechenden Klassen aus dem Lernraum hinzugefügt werden. Ansonsten werden keine Punkte vergeben.
- Einige Hausaufgaben müssen im Spiel Codescape gelöst werden. Klicken Sie dazu im Lernraum rechts im Block "Codescape" auf den angegebenen Link. Diese Aufgaben werden getrennt von den anderen Hausaufgaben gewertet.

Aufgabe 3 (Programmierung): (26 Punkte)

In dieser Aufgabe geht es um Datenstrukturen in Java. Ziel ist es, einen Stack[1] mittels Arrays zu implementieren. Ein Stack ist eine Datenstruktur, welche einen Stapel (oder "Keller") von Elementen darstellt. Mit der Methode `push()` kann ein Element oben auf den Stapel hinzugefügt werden, mit `pop()` hingegen wird das oberste Element entfernt.
Hierfür soll die bereitgestellte Klasse `SimpleIO` genutzt werden. Um einen String `str1` in einem Fenster auszugeben, nutzen Sie `SimpleIO.output(str1)`. Um einen Wert vom Typ `String` einzulesen, benutzen Sie `SimpleIO.getString("Geben Sie ein zu speicherndes Element ein:")`.
Schreiben Sie ein Java-Programm, welches einen solchen Stack darstellt. Hierzu sollen Sie die Klasse `OurStack` um die fehlenden Methoden ergänzen. Ihr Stack soll ausschließlich `String`s als Elemente beinhalten und besteht aus den folgenden zwei Attributen:

- `stack`: Ist ein `String`-Array, welches die Elemente des Stacks beinhaltet. Initial hat das Array den Wert `null`.
- `currentSize`: Speichert die aktuelle Anzahl der Elemente des Stacks. Daher hat die Variable `currentSize` initial den Wert `0`.

Implementierten Sie nun die folgenden Methoden in der bereitgestellten Klasse `OurStack`:

- `push()`: Wenn im Array, welches den Stack repräsentiert, noch Platz ist (also `currentSize` $<$ `stack.length`), dann soll ein String eingelesen werden und an die erste freie Stelle im Array gespeichert werden. Ist das Array jedoch voll, dann soll eine geeignete Meldung ausgegeben werden. Das Array wird in diesem Fall nicht verändert.
- `pop()`: Wenn sich im Stack mindestens ein Element befindet (also `currentSize` $>$ 0), dann soll das zuletzt hinzugefügte Element entfernt werden (`currentSize` soll also um 1 verringert werden). Befindet sich kein Element im Stack, dann soll nichts passieren.
- `clear()`: Alle Elemente sollen gelöscht werden (es soll also danach `currentSize` $= 0$ gelten).
- `setSize(int size)`: Es soll ein neues Array der Größe `size` erstellt werden, die ersten (d.h. zuerst eingefügten) min(`currentSize`, `size`)-Elemente kopiert werden und schließlich das ursprüngliche Array `stack` durch das neue Array ersetzt werden. Ist der Parameter `int size` negativ, so kann sich Ihre Methode beliebig verhalten.
- `print()`: Hier sollen die Elemente des Stacks durch Kommata getrennt ausgegeben werden. Hierbei sollen die zuerst eingefügten Elemente auch zuerst ausgegeben werden. Wenn der Stack leer ist, dann soll hingegen "Stack ist leer." ausgegeben werden.

In der Java-Datei `OurStack.java` ist bereits die statische Methode `main` bereitgestellt. Hier werden obige Methoden verwendet, um ein `OurStack`-Objekt zu verändern. Dazu werden solange Operationen mit `SimpleIO` eingelesen, bis `STOP` eingeben wurde. Sind Ihre Methoden korrekt implementiert, dann könnte ein Ablauf des Programms nun z.B. so aussehen:

```
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
SETSIZE
Bitte geben Sie die (nicht negative) Groesse ein:
1
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PUSH
Geben Sie ein zu speicherndes Element ein:
EINS
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PUSH
Stack ist voll.
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
SETSIZE
```

[1] https://de.wikipedia.org/wiki/Stapelspeicher

```
Bitte geben Sie die (nicht negative) Groesse ein:
2
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PUSH
Geben Sie ein zu speicherndes Element ein:
ZWEI
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PRINT
Stack: EINS,ZWEI
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
POP
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PRINT
Stack: EINS
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
PUSH
Geben Sie ein zu speicherndes Element ein:
DREI
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
SETSIZE
Bitte geben Sie die (nicht negative) Groesse ein:
1
Bitte geben Sie eine Operation (PUSH,POP,CLEAR,SETSIZE,PRINT,STOP) ein:
STOP
Stack: EINS
```

Hinweise:

- Legen Sie die bereitgestellte Datei `SimpleIO.java` im gleichen Verzeichnis wie Ihre Lösung ab. Dann findet Java diese automatisch.
- Sie können `Math.min(x,y)` und `Math.max(x,y)` verwenden, um das Minimum bzw. Maximum von `x` und `y` zu berechnen.
- Im Moodle-Lernraum steht die Java-Datei `OurStack.java` zum Download zur Verfügung. Vervollständigen Sie die Implementierung in dieser Klasse!

Aufgabe 5 (Verifikation mit Arrays): (16 + 8 = 24 Punkte)

Gegeben sei folgendes Java-Programm P:

$\langle$ `a.length > 0` $\rangle$ (Vorbedingung)

```
n = a.length;
i = 0;
res = true;
while(i < n - 1) {
  if(a[i] > a[i+1]) {
    res = false;
  }
  i = i + 1;
}
```

$\langle$ `res` $= \forall\, 0 \leq k < \mathtt{n}-1 : \mathtt{a}[k] \leq \mathtt{a}[k+1]$ $\rangle$ (Nachbedingung)

a) Vervollständigen Sie die folgende Verifikation des Algorithmus im Hoare-Kalkül, indem Sie die unterstrichenen Teile ergänzen. Hierbei dürfen zwei Zusicherungen nur dann direkt untereinander stehen, wenn die untere aus der oberen folgt. Hinter einer Programmanweisung darf nur eine Zusicherung stehen, wenn dies aus einer Regel des Hoare-Kalküls folgt.

Beachten Sie bei der Anwendung der "Bedingungsregel 1" mit Vorbedingung φ und Nachbedingung ψ, dass auch $\varphi \wedge \neg B \implies \psi$ gelten muss. D. h. die Nachbedingung ψ der `if`-Anweisung muss aus der Vorbedingung φ der `if`-Anweisung und der negierten Bedingung $\neg B$ folgen. Geben Sie beim Verwenden der Regel einen entsprechenden Beweis an.

Hinweise:

- Gehen Sie wieder bei allen Aufgaben zum Hoare-Kalkül davon aus, dass keine Integer-Überläufe stattfinden, d.h., behandeln Sie Integers als die unendliche Menge $\mathbb{Z}$.
- Sie dürfen beliebig viele Zusicherungs-Zeilen ergänzen oder streichen. In der Musterlösung werden allerdings genau die angegebenen Zusicherungen benutzt.
- Bedenken Sie, dass die Regeln des Kalküls syntaktisch sind, weshalb Sie semantische Änderungen (beispielsweise von $x+1 = y+1$ zu $x = y$) nur unter Zuhilfenahme der Konsequenzregeln vornehmen dürfen.
- Geben Sie jeweils eine kurze Begründung an, warum die Konsequenzregeln korrekt angewandt wurden. D.h. beweisen Sie, dass aus der oberen Zusicherung die untere folgt, wenn diese direkt untereinander stehen.
- Der Ausdruck `res` $= \forall 0 \leq k < i : \mathtt{a}[k] \leq \mathtt{a}[k+1]$ hat den Wert `true`, wenn für jede natürliche Zahl $k \in \{0, 1, \ldots, i-1\}$ jeweils $\mathtt{a}[k] \leq \mathtt{a}[k+1]$ gilt, sonst hat der Ausdruck den Wert `false`. Ist i eine ganze Zahl kleiner als 1, so hat der Ausdruck ebenfalls den Wert `true`. Die Nachbedingung in unserem Beispiel besagt also, dass `res` genau dann den Wert `true` hat, wenn das Array aufsteigend sortiert ist.

$\langle$`a.length > 0`$\rangle$

$\langle$_____$\rangle$

```
n = a.length;
```

$\langle$_____$\rangle$

```
i = 0;
```

$\langle$_____$\rangle$

```
res = true;
```

$\langle$_____$\rangle$

$\langle$_____$\rangle$

```
while (i < n - 1) {
```

$\langle$_____$\rangle$

$\langle$_____$\rangle$

```
    if(a[i]>a[i+1]) {
```

$\langle$_____$\rangle$

$\langle$_____$\rangle$

```
        res = false;
```

$\langle$_____$\rangle$

```
    }
```

$\langle$_____$\rangle$

```
    i = i + 1;
```

$\langle$_____$\rangle$

```
}
```

$\langle$_____$\rangle$

$\langle \texttt{res} = \forall\, 0 \leq k < \texttt{n} - 1 : \texttt{a}[k] \leq \texttt{a}[k+1] \rangle$

b) Untersuchen Sie den Algorithmus P auf seine Terminierung. Für einen Beweis der Terminierung muss eine Variante angegeben werden und unter Verwendung des Hoare-Kalküls die Terminierung unter der Voraussetzung `a.length > 0` bewiesen werden.

Geben Sie auch bei dieser Teilaufgabe einen Beweis für die Aussage $\varphi \wedge \neg B \implies \psi$ bei der Anwendung der "Bedingungsregel 1" an.

Aufgabe 6 (Deck 3): (Codescape)

Lösen Sie die Missionen von Deck 3 des Codescape Spiels. Ihre Lösung für die Codescape Missionen wird nur dann für die Zulassung gezählt, wenn Sie Ihre Lösung vor der einheitlichen Codescape Deadline am Freitag, den 24.01.2025, um 23:59 Uhr abschicken.