hochschule mannheim

FAKULTÄT FÜR INFORMATIK

Prof. Dr. Peter Knauber

PR1 – Programmieren 1 | Bachelor-Studiengang Informatik | WS2019/2020

Übungsblatt 6: Strings, Zeichen, Schleifen

Ausgabe: 11.11.19
Abgabe: 20.11.19 10:00 Uhr elektronisch mittels Git

Aufgabe 1

Erstellen Sie eine Klasse `Reverser`, mit der Sie einen Satz umdrehen können. Fügen Sie Ihrer Klasse eine Methode `String reverseSentence( String sentence )` hinzu, welche die Reihenfolge der Wörter im übergebenen String `sentence` umdreht und das Ergebnis wiederum als String zurückliefert. Beispiele:

`"Hello world"`

soll umgewandelt werden in

`"world Hello"`.

`"May the force be with you"`

soll umgewandelt werden in

`"you with be force the May"`.

Hinweise:

- Sie können davon ausgehen, dass die eingegebenen "Sätze" keine Satzzeichen oder Sonderzeichen enthalten. Sie können ebenfalls davon ausgehen, dass nicht `null` übergeben wird.
- Wenn die Eingabe mehrere aufeinanderfolgende Leerzeichen enthält, soll die Ausgabe dafür nur ein einzelnes Leerzeichen enthalten.
- Aus der Klasse `String` dürfen nur die Methoden `length`, `charAt` und `equals` verwendet werden, *keine* anderen!
- Es kann hilfreich sein, sich eigene Hilfsmethoden zu schreiben.

Erstellen Sie eine `main`-Methode, in der man Eingaben machen kann, um die Funktion zu Testzwecken aufzurufen.

Aufgabe 2

Erstellen Sie ein Java-Programm (Klasse "DividersArrayResult"), das alle Teiler einer *natürlichen Zahl* berechnet.
In der `main`-Methode soll der Benutzer aufgefordert werden, eine ganze Zahl einzugeben. Nachdem er seine Zahl eingetippt hat, wird eine Methode `calculateDividers` aufgerufen, welche die eingegebene Zahl als Parameter erhält. Diese Methode soll alle Teiler des übergebenen Parameters in ein Array von ganzen Zahlen der Länge 500 schreiben. Sie können sich darauf verlassen, dass `calculateDividers` nicht mit Zahlen größer 500 aufgerufen wird.

Nachdem alle Teiler in das Array geschrieben wurden, liefert `calculateDividers` das Array als Ergebnis an `main` zurück. In `main` soll der sinnvolle Inhalt des Arrays (d.h. nur die gefundenen Teiler) in einer Zeile ausgegeben werden. Dabei werden die enthaltenen Zahlen jeweils durch ein Leerzeichen getrennt.
Falsche Eingaben sollen in `main` durch eine Fehlermeldung "Eingabe ungültig" signalisiert werden. Das Programm endet nach Ausgabe der Teiler bzw. der Fehlermeldung.

Sie können sich darauf verlassen, dass nur ganze Zahlen eingegeben werden.
Wählen Sie vernünftige Namen für Ihre Variablen!

Aufgabe 3

Erstellen Sie ein Java-Programm (Klasse "SearchInRandomNumbers"), dessen `main`-Methode zwei ganze Zahlen `numberCount` und `numberToSearch` abfragt.

Die Zahl `numberCount` wird an eine Methode `generate` übergeben, die ein Array für ganze Zahlen mit `numberCount` Elementen anlegt, dieses Array mit Zufallszahlen zwischen 1 und 1000 (beide inklusive) füllt und das befüllte Array als Ergebnis an `main` zurückliefert.
Klarstellung: Ist die Zahl `numberCount` kleiner als 0, wird in der Methode `generate` eine `PRException` ausgelöst.
Halten Sie sich für die Erzeugung der Zufallszahlen an das Muster mit den Lottozahlen aus der Vorlesung.

Die `main`-Methode übergibt das erzeugte Array und die zweite eingegebene Zahl `numberToSearch` an eine zweite Methode `searchAll`. Diese Methode sucht *alle* Positionen von `numberToSearch` im übergebenen Zufallsarray, legt ein neues Array an, dessen Länge genau der Anzahl der gefundenen Positionen entspricht, befüllt dieses Array mit den gefundenen Positionen und liefert es als Ergebnis zurück. Die `main`-Methode gibt dann alle Positionen auf der Konsole aus.
Klarstellung: Falls die Zahl `numberToSearch` nicht gefunden wird, wird ein Positionsarray der Länge 0 erzeugt und nichts ausgegeben.

Danach übergibt `main` das Array mit den Zufallszahlen und die Zahl `numberToSearch` an eine dritte Methode `searchLast`. Diese sucht die *letzte* Position von `numberToSearch` und liefert diese Position als Ergebnis zurück. In der `main`-Methode wird diese Position auf der Konsole ausgegeben.
Klarstellung: Falls die Zahl `numberToSearch` nicht gefunden wird, wird in der Methode `searchLast` eine `PRException` ausgelöst.

Allgemeines

- Legen Sie für die Bearbeitung dieses Übungsblattes ein Paket namens `uebung06` an, in dem Sie Ihre Klassen anlegen.
- Sie können sich darauf verlassen, dass nur ganze Zahlen eingegeben werden (keine Buchstaben oder Sonderzeichen).
- In Moodle finden Sie Testklassen `...1stTest`, mit der Sie prüfen können, ob Ihre Methoden korrekt definiert sind und zumindest die einfachsten Eingaben korrekt behandelt werden.
- Erlaubt sind `MakeItSimple`-Funktionen (keine nicht besprochene Funktionalität aus der Java-Standard-Bibliothek) und das bisher erworbene Wissen aus den PR1-Vorlesungen. Sie müssen alle(!) von Ihnen verwendeten Konstrukte der Sprache sowie alle verwendeten Methoden, die nicht aus der Hilfsbibliothek `MakeItSimple` stammen, gut erklären und nötigenfalls im Testat selbst programmieren können!
- Sie geben ab, indem Sie vor Ende der Abgabefrist Ihr Projekt mit den lauffähigen Programmen des Übungsblattes in das Repository auf *wilma* pushen. Achten Sie darauf, ob Sie die Kontroll-E-Mail bekommen! Die letzte hochgeladene Version Ihres Projekts wird gewertet. Andere Abgaben, ob elektronisch oder auf Papier, zählen als **nicht abgegeben**!
- Die Aufgaben sind in Eclipse zu bearbeiten und beim Testat vorzuführen.
- Nutzen Sie die Übungsstunde, um eventuelle Fragen zur Vorlesung, zur Übung oder allgemein zum Studium zu stellen! Die Betreuer sind da, um Ihnen solche Fragen zu beantworten und Ihnen bei Bedarf zu helfen!