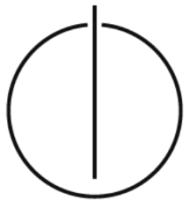

TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK
Praktikum: Grundlagen der Programmierung

Programmierpraktikum

Woche 05 (24.11.2016)

Stefan Berktold
s.berktold@tum.de

Präsenzaufgaben

Heutige Übersicht

Aufgabe 5.1 ***Programmieraufgabe: Arrays***
Grundlagen von Arrays (insb. Iterieren/Indices)

Aufgabe 5.2 ***Programmieraufgabe: Palindrom***
Palindromerkennung mithilfe von Arrays über `chars`

Aufgabe 5.3 ***Programmieraufgabe: Matrizen und Vektoren***
Rechnen mit als Arrays dargestellten Vektoren: Vektor-/Matrizenmultiplikation

Hausaufgaben

Abgabe: 28.11.2016 05:00 Uhr

Aufgabe 5.4 [20+*2*] ***Programmieraufgabe (MiniJava**, erweitert**)***

Spiel: Linja basierend auf einem zweidimensionalen Array (Spielfeld)

Aufgabe 5.5 [*3 Bonus*] ***Programmieraufgabe (MiniJava**, erweitert**)***

Cäser-Verschlüsselung jetzt funktional

- bestehende Lösung verwenden und Code in `shift` *auslagern*
(vgl. Lösungsvorschlag zu A 4.6 auf tutor16.stecrz.de)

Aufgabe 5.1

Grundlegende Array-Methoden (Auszug)

In einer Klasse Arrays sollen die folgenden Methoden implementiert werden:

- `public static void print(int[] feld)`
- `public static void minUndMax(int[] feld)`
- `public static int[] invertieren(int[] feld)`
- `public static int[] schneiden(int[] feld, int laenge)`
- `public static int[] linearisieren(int[][] feld)`

Die vollständige Angabe (PDF) gibt es auf *Moodle* und wird auch für nachfolgende Aufgaben benötigt. Es dürfen nur die dort im Anhang genannten Java-Bausteine verwendet werden. Ein Code-Gerüst (Arrays.java), mit dem Sie Ihre Methoden testen können, finden Sie auf tutor16.stecrz.de.

Aufgabe 5.1 – Lösung

Grundlegende Array-Methoden

Meinen (inoffiziellen) Lösungsvorschlag gibt es unter http://tutor16.stecrz.de.

Die offiziellen Lösungen zu allen Präsenzaufgaben sind voraussichtlich ab

25.11.2016 18:00 Uhr

auf *Moodle* verfügbar (und dürfen nicht vorab ausgehändigt werden).

Aufgabe 5.2

Palindrom (Auszug)

- Schreiben Sie eine Methode `public static String toLowerCase(String input)`, die als Rückgabewert einen String liefert, der die gleiche Zeichenfolge wie `input` ist, wobei alle Großbuchstaben `A` bis `Z` in den entsprechenden Kleinbuchstaben umgewandelt sind.
- Schreiben Sie eine Methode `public static char[] toCharArray(String input)`, die den String `input` in ein `char`-Array umwandelt und dieses als Rückgabewert zurückgibt. Dabei gilt, dass der erste Buchstabe an der ersten Position im Array steht, der zweite Buchstabe an der zweiten Position usw.
- Schreiben Sie eine Methode `public static boolean isPalindrome(char[] input)`, die das Array `input` mit Hilfe einer Schleife durchläuft und bestimmt, ob es sich bei der *gesamten* Zeichenkette um ein Palindrom handelt. Der Rückgabewert is `true`, wenn die Zeichenkette ein Palindrom ist, ansonsten `false`. Überlegen Sie sich, ob Sie das Array in seiner vollen Länge durchlaufen müssen, oder ob es reicht, über die Hälfte zu iterieren.

Aufgabe 5.2 – Lösung

Palindrom

Meinen (inoffiziellen) Lösungsvorschlag gibt es unter http://tutor16.stecrz.de.

Die offiziellen Lösungen zu allen Präsenzaufgaben sind voraussichtlich ab

25.11.2016 18:00 Uhr

auf *Moodle* verfügbar (und dürfen nicht vorab ausgehändigt werden).

Aufgabe 5.3

Matrix- und Vektormultiplikation (Auszug 1/2)

1. Multiplikation zweier Vektoren a und b:

```
public static int vecvecmul(int[] a, int[] b)
```

$$\begin{pmatrix} a_1 & a_2 & \dots & a_n \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \sum_{j=1}^{n} a_j \cdot b_j$$

2. Multiplikation einer Matrix a mit einem Vektor b:

```
public static int[] matvecmul(int[][] a, int[] b)
```

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_m \end{pmatrix}$$

wobei die Einträge c_i gegeben sind durch

$$c_i = \sum_{j=1}^{n} a_{ij} \cdot b_j.$$

Aufgabe 5.3

Matrix- und Vektormultiplikation (Auszug 2/2)

3. Transposition einer Matrix a:

```
public static int[][] transpose(int[][] a)
```

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}^{\top} = \begin{pmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{mn} \end{pmatrix}$$

4. Multiplikation zweier Matrizen a und b:

```
public static int[][] matmatmult(int[][] a, int[][] b)
```

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \cdot \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & \dots & b_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nk} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1k} \\ c_{21} & c_{22} & \dots & c_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{n2} & \dots & c_{mk} \end{pmatrix}$$

wobei die Einträge c_{ij} gegeben sind durch $c_{ij} = \sum_{l=1}^{n} a_{il} \cdot b_{lj}$

Aufgabe 5.3 – Lösung

Matrix- und Vektormultiplikation

Die offiziellen Lösungen zu allen Präsenzaufgaben sind voraussichtlich ab

25.11.2016 18:00 Uhr

auf *Moodle* verfügbar (und dürfen nicht vorab ausgehändigt werden).

Tipps zu Hausaufgabe 5.4

Linja (1/2)

1. **gueltigeEingabe**() – soll lediglich true/false zurückgeben (return), je nachdem ob stein im Wertebereich von spieler liegt (Spieler 1: 1 bis 12, Spieler -1: -12 bis -1)
2. **findeStein**() – gibt die Position von Stein in einem Array der Form {x, y} zurück (return), wobei logisch betrachtet x dem Spalten- und y dem Zeilenindex entspricht
3. **steineInReihe**() – gibt die Anzahl an Steinen in Zeile reihe zurück (return)
4. **zaehlePunkte**() – wird am Ende des Spiels aufgerufen und soll die Punkte für beide Spieler berechnen und ausgeben (write). Zu beachten sind neben Steinen auf dem Feld (-5 bis 5 Punkte) auch diejenigen Steine, die das Ziel bereits erreicht haben, aber nicht mehr angezeigt werden (laut Aufgabenstellung) (5 Punkte).
5. **spielende**() – gibt true zurück (return), falls das Spiel beendet ist bzw. beendet werden soll (gdw. beide Spieler komplett aneinander vorbeigezogen), sonst false

Tipps zu Hausaufgabe 5.4

Linja (2/2)

6. **setzeZug**() – zieht mit dem Spielstein stein um weite Felder nach vorne (falls vorwaerts==true, sonst nach hinten), sofern der Zug erlaubt ist (d. h. Spieler geht nicht weiter zurück als seine Startposition, Zielreihe ist noch nicht voll, Stein existiert überhaupt noch). Was „nach vorne“ bedeutet hängt natürlich vom Spieler ab (negative Spielsteine von unten nach oben, positive von oben nach unten). Welche 0 (leeres Feld) der Zielreihe mit dem Spielstein stein besetzt wird ist dabei egal (nimm einfach die erste gefundene freie Position).
7. **spielerZieht**() – führt den Spielablauf für spieler durch; d. h. erst Anfangszug, dann Folgezug (evtl. gefolgt von Bonuszug). Die Eingabe (read) des Spielsteins muss überprüft werden (1. Ungültige Eingabe, 2. Zug nicht möglich); evtl. Fehlermeldung ausgeben und neu eingeben lassen!
8. **main**() – beide Spieler abwechselnd ziehen lassen bis Spiel vorbei, Punkte ausgeben.

Tipps zu Hausaufgabe 5.4

Der *funktion*ierende Cäsar

1. Wenn deine Lösung zu **Aufgabe 4.6** korrekt war, verwende diese als Vorlage, anderenfalls Lösungsvorschläge auf *Moodle* oder tutor16.stecrz.de nachvollziehen.
2. Die Methode **`shift`** soll ein einzelnes Zeichen um `k` (Schlüssel) verschieben. Diesen Teil hat man bereits für jedes Zeichen in der Lösung zu Aufgabe 4.6 ausgeführt. Achte nur darauf, dass `shift` auch für alle negativen `k` funktionieren muss.
3. Die Methode **`encrypt`** wird nun die Methode `shift` aufrufen und so den gesamten String verschlüsseln. Orientiere dich an deiner alten Lösung.
4. Teilaufgabe 3 nicht vergessen: String aus der Aufgabenstellung mit `write` korrekt entschlüsseln, sodass der entschlüsselte String ausgegeben wird (Aufgabe ≈ Key finden).