

- i) Schreiben Sie eine Methode `int[][] erzeugeFeldEinfach(int n, int m)`, die ein zweidimensionales Feld mit n Zeilen und m Spalten erzeugt, dieses Feld mit dem folgenden Muster füllt und zurückgibt:

0	0	1	1	0	0
0	0	1	1	0	0
1	1	0	0	1	1
1	1	0	0	1	1
0	0	1	1	0	0

D.h. es sollen alternierend 2x2 Blöcke mit 0 und 1 gefüllt werden, wobei die 0. und 1. Zeile mit einem Block mit 0 beginnt, die 2. und 3. Zeile mit einem Block von 1 usw. Sie dürfen die Methode `erzeugeFeld` aus Teil ii) **nicht** verwenden!

- ii) Schreiben Sie nun eine Methode `int[][] erzeugeFeld(int n, int m, int s, int t)`, die wie oben ein Feld mit n Zeilen und m Spalten erzeugt, es mit alternierenden Blöcken mit 0 und 1 füllt (s.o.) und zurückgibt. Diesmal sollen die einzelnen Blöcke aber s Zeilen und t Spalten besitzen. Mit s=2 und t=2 erhält man also wieder das Muster aus Teil i).

Sie dürfen **keine** Klassen- oder Objektattribute oder Methoden aus der Java-Klassenbibliothek benutzen. Sie können aber weitere eigene (Hilfs-)Methoden hinzufügen.

- b) Schreiben Sie die Java-Methode `public void fuelleMatrix()`, die das Feld `data` mit einem Muster ausfüllt. Dieses Muster soll wie im folgenden Beispiel eines (5,5)-Arrays aussehen:

<b>0</b>	1	2	3	4
-1	<b>0</b>	1	2	3
-2	-1	<b>0</b>	1	2
-3	-2	-1	<b>0</b>	1
-4	-3	-2	-1	<b>0</b>

Hinweis:

Ihre Methode soll allgemein für alle Größen des Feldes funktionieren, nicht nur für die Größe des Beispiels.

```
public void fuelleMatrix()
{
```

- a) Schreiben Sie die Java-Methode `public void fuelleFeld(char oben, char unten)`, die das Feld `feld` mit einem Muster ausfüllt. Dieses Muster soll wie im folgenden Beispiel eines (5,5)-Arrays aussehen, wenn als Parameter für oben 'o' und für unten 'u' verwendet wurde:

o	o	o	o	o
o	o	o	o	o
=	=	=	=	=
u	u	u	u	u
u	u	u	u	u

Die mittlere Zeile wird immer mit '=' gefüllt.

Hinweis: Alle Methoden sollen allgemein für alle Größen des Feldes funktionieren, nicht nur für die Größe des Beispiels.

```
public void fuelleFeld(char oben, char unten)
{
```

- b) Schreiben Sie die Java-Methode `public void fuelleOberesDreieck(char fuellzeichen)`, die das Feld `feld` oberhalb der Mittellinie mit einem auf dem Kopf stehenden Dreieck füllt. (Alle anderen Array-Elemente sollen unverändert bleiben.) Als Füllzeichen soll der übergebene Parameter `fuellzeichen` verwendet werden. Für ein (7,7)-Array, das zuvor durch Aufruf von `fuellerFeld` wie in a) gezeigt gefüllt wurde, ergibt sich durch anschließenden Aufruf von `fuellerOberesDreieck('A')`:

o	A	A	A	A	A	o
o	o	A	A	A	o	o
o	o	o	A	o	o	o
=	=	=	=	=	=	=
u	u	u	u	u	u	u
u	u	u	u	u	u	u
u	u	u	u	u	u	u

```
public void fuelleOberesDreieck(char fuellzeichen)
{
```

- c) Schreiben Sie die Java-Methode `public void spiegele(char zuSpiegelndesZeichen)`, die das Feld `feld` oberhalb der Mittellinie nach dem zu spiegelnden Zeichen durchsucht. Jedes gefundene Zeichen wird an der Mittellinie gespiegelt. (Alle anderen Array-Elemente bleiben unverändert.) Angewendet auf das Beispiellarray aus Teil b) ergibt sich mit 'A' als an der Mittellinie zu spiegelndem Zeichen:

o	A	A	A	A	A	o
o	o	A	A	A	o	o
o	o	o	A	o	o	o
=	=	=	=	=	=	=
u	u	u	A	u	u	u
u	u	A	A	A	u	u
u	A	A	A	A	A	u

```
public void spiegele(char zuSpiegelndesZeichen)
{
```

In dieser Aufgabe soll ein zweidimensionales Array mit  $n$  Zeilen und  $2n-1$  Spalten mit folgendem Muster (ein großes A) ausgefüllt werden (**das folgende Array ist ein Beispiel, Ihr Algorithmus soll für allgemeine Größen  $n$  funktionieren**):

				*				
			*		*			
		*	*	*	*	*		
	*						*	
*								*

Die freien Flächen werden mit Leerzeichen ( ' ' ) ausgefüllt.

Füllen Sie ein zweidimensionales Array mit  $n$  Zeilen und  $n$  Spalten mit den folgend beschriebenen Mustern. Das folgenden Arrays sind Beispiele, Ihre Algorithmen soll für allgemeine Größen  $n$  ( $n$  ungerade) funktionieren. Die freien Flächen werden mit Leerzeichen ( ' ' ) ausgefüllt.

- a) **Muster 1:** Implementieren Sie die Methode **fülleArrayMitA**.

	A		A	
A		A		A
	A		A	
A		A		A
	A		A	

- b) **Muster 2:** Implementieren Sie die Methode **fülleArrayMitBC**. Überlegen Sie zuerst, wie Sie die B-Strecken erzeugen. Überlegen Sie dann, wie Sie die C-Strecken erzeugen.

			B		
		B		C	
	B				C
B					B
	C				B
		C		B	
			B		

Implementieren Sie die Methode `void createPascal(int size)`, die ein Pascalsches Dreieck der Größe `size` erstellt und ausgibt. Ein Pascalsches Dreieck wird folgendermaßen konstruiert: An der obersten Stelle steht eine 1. An allen anderen Stellen steht je die Summe der beiden darüberstehenden Zahlen. Fehlt einer der darüberstehenden Summanden (d.h. an den Rändern), wird eine 1 eingetragen.

- Erzeugen und füllen Sie innerhalb der Methode `createPascal` ein entsprechendes nicht-rechteckiges Array. Im Beispiel sehen Sie das Array, das mit dem Aufruf `createPascal(5)` erzeugt wird.
- Geben Sie anschließend innerhalb der Methode `createPascal` das Dreieck aus.

```

      1
    1 1
  1 2 1
1 3 3 1
1 4 6 4 1
Pascalsches Dreieck

```

```

  1
 1 1
1 2 1
1 3 3 1
1 4 6 4 1
Zu erzeugendes Array

```

```

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
Ausgabe

```

```
private static void createPascal(int size) //Annahme: size>=1
{

```

Schreiben Sie die beiden in Aufgaben a) und b) angegebenen Java-Methoden für

- ein gegebenes **quadratisches** Feld `char[][] feld`
- mit einer **ungeraden** Anzahl von Zeilen und Spalten.

- Schreiben Sie die Java-Methode

**public void fuehleFeld(char[][] feld, char links, char rechts),**

die ein Muster wie unten im Beispiel angegeben erzeugt, wenn als Parameter für links 'L' und für rechts 'R' verwendet wurde. Die mittlere Spalte wird immer mit 'X' gefüllt.

L	L	X	R	R
L	L	X	R	R
L	L	X	R	R
L	L	X	R	R
L	L	X	R	R

Beispielfeld mit 5 Zeilen  
und Spalten

```
public void fuehleFeld(char[][] feld, char links, char rechts)
{

```

- b) Schreiben Sie die Java-Methode **public void diagonale(char[][] feld, char zeichen)**, die ein Muster wie unten angegeben erzeugt, wenn als Parameter für zeichen 'Y' verwendet wurde. In den restlichen Einträgen soll das leere Zeichen ' ' erscheinen:

Y		Y		Y
	Y		Y	
Y		Y		Y
	Y		Y	
Y		Y		Y

Beispielfeld mit 5 Zeilen  
und Spalten

```
public void diagonale(char[][] feld, char zeichen)
{
```