University of Konstanz
Department of Computer and Information Science
Chair for Data Analysis and Visualization
Dr. Johannes Fuchs

Due: Tuesday, 23.11.2021, 10:00

Task 1 (2 points):

Write a method to check, whether a `char` array is a palindrome. A palindrome is a sequence of characters, which reads the same backward as forward.
Examples are: `'m' 'a' 'd' 'a' 'm'` or `'a' 'n' 'n' 'a'`

There are multiple solutions to solve the problem. One possibility is to compare the first character and the last character of the sequence. If they are equal, you should continue to compare the second character and the second last character and so on. Once you discover a pair which is not equal you should return `false`. If you run through half of the array having only equal pairs you should return `true`.

```
public static boolean testPalindrome(char[] input)
```

Task 2 (1 point):

Write a method to sort an integer array. Please, implement the bubble sort algorithm introduced in the lecture. Your method should return the sorted `int` array.

```
public static int[] bubbleSort(int[] array)
```

Task 3 (1 point):

Write a method, which returns the median value of a given integer array. If the array length is even, the median cannot be returned directly. In this case, your method should add the two arbitrary values in the middle and then divide the result by 2. If the array is empty, your method should return 0.

```
public static double getMedian(int[] array)
```

Task 4 (2 points):

Write a method to delete an integer number from an `int` array.
First, you have to check, whether the array contains this number.

- If the array does not contain the number, then you should just return the initial array.
- If the array contains the number continue with the second step:

Second, you have to instantiate a new array, which is 1 element shorter than the original array.
Third, you have to copy all elements except for the searched number to the new array.
Forth, you have to return the new array.

```
public static int[] deleteElement(int[] a, int element)
```

Task 5 (2 points):

This task continues Task 3 from Assignment 3. Go back to it to read up on the basics.
Use your implemented method to calculate multiple bets of a single person. Your new implementation should take a 2-dimensional array (like the table) and return the total amount of points gained. The data should be arranged as shown in the table below.

Given the following table the result should be 4 points.

Home	Guest	homeBet	guestBet
3	2	3	2
1	1	1	0
2	2	1	1

For the first bet the actual result was 3:2 and the bet was 3:2 (3 Points), the second bet 1:1 / 1:0 (0 points) and the third bet 2:2 / 1:1 (1 point, correct tendency).

```
public static int soccerBets(int[][] bets)
```

IMPORTANT HINT: Remember that you already programmed the method `soccerBet`! If you follow the project structure outlined in the previous exercise you can reuse it without having to write it again. This is a very important concept in programming:

DON'T REPEAT YOURSELF

For example, if you have the package `assignment03` with the class `Solution03` which has a static method `soccerBet` you can use `assignment03.Solution03.soccerBet()` to use it for this assignment.

STYLE POINTS

Style (2 points):

To comply with the required code style you must have:

1. named your variables appropriately
2. used the different data types in a meaningful way.
3. used indention correctly
4. commented your code appropriately e.g., JavaDoc-comments (description, parameters, return values, …)
5. used loops for repeating code segments

PROJECT MILESTONE 2

Milestone 2 is due till 30.11.2021:

Task 1: Providing Data

Implement a two-dimensional array, which contains data points in the first dimension and in the second dimension the corresponding x- and y-coordinates. Four data points stored in such a two-dimensional array could look like this:

```
double[][] points = {{1, 1}, {2, 2}, {1, 2}, {2, 1}};
```

Task 2: Extend the Class `Utilities.java`

Given your knowledge about loops and arrays, we will add some more functionality to the `Utilities.java` class.

Implement a method “`getPointOfIntersection`”, which takes four arrays as input. Each array corresponds to one data point with the first entry of each array being the x-coordinate and the second entry representing the y-coordinate. Given a line between point1 and point2 and another line between point3 and point4 calculate the point of intersection and return this point as an array (first entry is the x-coordinate, second entry is the y-coordinate). The method should have the following signature:

```
public static double[] getPointOfIntersection(double[] point1,
double[] point2, double[] point3, double[] point4)
```

Task 3: Check your Code

Given the example data points from Task1 and the new method implemented in Task2, you can simply check your code by looking at the result.

Example code:

```
double[][] points = {{1, 1}, {2, 2}, {1, 2}, {2, 1}};

double[] intersection = getPointOfIntersection(points[0], points[1],
points[2], points[3]);

System.out.println("Intersection: " + intersection[0] + "   " +
intersection[1]);
```

Result:

```
Intersection: 1.5  1.5
```