

Lab lecture exercises – Week 10

1. Write a method `public static int[] reverse(int[] a)` that reverses a one dimensional array (e.g. $\{1,2,3,4\}$ goes to $\{4,3,2,1\}$).
2. Write a method `public static double min(double[][] a)` that computes the minimum of a two-dimensional array. For an empty array an exception should be thrown.
3. Write a method `public static double[] copy(double[] a)` that copies a one-dimensional array of double. Likewise write a method that copies a two-dimensional array: `public static double[][] copy(double[][] a)`.

4. A magic square is an arrangement of the numbers $1, 2, \dots, n^2$ in an $n \times n$ matrix so that each number occurs exactly once and the sum of the numbers in each row and column as well as the two diagonals is always the same value. For an example find a magic square to the right.

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

Write a program that first reads in the size n of a magic square and then reads in the n^2 numbers (all from the command line). Your program should then check that the numbers form actually a magic square:

- Does each number in $1, 2, \dots, n^2$ occur exactly once?
- Are the conditions on the sums met?

[taken from Horstmann, Big Java, 4th edition p. 299.]

5. Write a class `Complex` with the two fields `private double realPart` and `private double complexPart`. Furthermore write a constructor, getters, and a `public String toString()` method as well as a method `public Complex add(Complex summand)` and a method `public Complex multiply(Complex factor)` for addition and multiplication, respectively.
6. Write a class `Name` with two fields `private String firstName` and `private String surname`. Assume that you have arrays of equal lengths, one containing strings representing first names, the other surnames. Write a method `public static Names[] makeNames(String[] firstNames, String[] surnames)`; to create a corresponding array of names.
7. Write a method `public static double convert(double amount, String fromUnit, String toUnit)` that can convert temperatures (degrees Fahrenheit to degrees Celsius and vice versa) and distances (miles, feet, inches, kilometres, metres, centimetres, millimetres into each other). Write code that is extensible so that you can easily add conversions between weights (e.g., pounds, ounces, kilograms, grams). The method should throw a suitable exception if called with incompatible units (i.e., if the `fromUnit` is incompatible with the `toUnit`).

- Assume that all the lengths of the strings are between 1 and 20 and the array has a length of at most 50. Write a method that in such cases prints the lengths in the following form (Example shows the lengths corresponding to quoted text in the next exercise from the Universal Declaration of Human Rights):

[illegible]

- All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.

Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status. . . .

<p>All human beings are born free and equal in dignity and rights. They are endowed with reason and conscience and should act towards one another in a spirit of brotherhood.</p>

Everyone is entitled to all the rights and freedoms set forth in this Declaration, without distinction of any kind, such as race, colour, sex, language, religion, political or other opinion, national or social origin, property, birth or other status. ...

11. Write an interface **Area** that contains the method signature **double area()**.

Write a class **Rectangle** that implements **Area** with field variables **private double length** and **private double width**. In particular, write a **public String toString()** method. Furthermore write a **Square** sub-class of **Rectangle**.

Write a class **Ellipse** that implements **Area**. It has two fields **private double majorRadius** and **private double minorRadius**. The area is given as **Math.PI * majorRadius * minorRadius**. Write a sub-class **Circle** of **Ellipse**. For **Circle** also write a method **private double circumference()**. (For a circle with radius r , the circumference is computed as $2 \cdot \pi \cdot r$.)