Exercises Lecture 1 (Sections 3.1-3.5)

Make sure to import Numpy to be able to use all its functionality.

```python
import numpy as np
```

Do not use for- or while-loops when answering the questions below.

Question 1

Create a 3×2 array M of ones, i.e., the matrix

```
[[1. 1.]
 [1. 1.]
 [1. 1.]]
```

Question 2

Create the array $x = [2, 4, 6, 8, ..., 100]$ once with `np.arange()`, and once with `np.linspace()`.

Question 3

Create the array $x = [-1, -0.9, \dots, -0.1, 0, 0.1, \dots, 0.9, 1]$.

Question 4

Consider the two-dimensional array below and answer the questions below. Recall that Python always starts counting at 0 when indexing.

```python
M = np.array([[1,1,1,1,1],[2,1,2,1,2],[2,1,2,1,2],
              [2,1,2,1,2],[2,1,2,1,2],[1,1,1,1,1]])

print(M)
```

```
[[1 1 1 1 1]
 [2 1 2 1 2]
 [2 1 2 1 2]
 [2 1 2 1 2]
 [2 1 2 1 2]
 [1 1 1 1 1]]
```

a) Index/access the odd-numbered rows ($1, 3$ and 5) of M. Your output should be

```
[[2 1 2 1 2]
 [2 1 2 1 2]
 [1 1 1 1 1]]
```

b) Index/access the rows and columns that form the submatrix consisting of elements equal to 2, i.e., obtain from M the matrix

```
[[2 2 2]
 [2 2 2]
 [2 2 2]
 [2 2 2]]
```

c) Index/access the submatrix consisting of the rows $0, 1, 5$ and columns $1, 3, 4$, i.e.,

```
[[1 1 1]
 [1 1 2]
 [1 1 1]]
```

Question 5

Consider the following array.

```python
x = np.array([3,6,4,5,5,5,1,4,2,9,6,7,11,10])

print(x)
```

```
[ 3  6  4  5  5  5  1  4  2  9  6  7 11 10]
```

a) Use a Boolean mask to access all element whose value is smaller or equal than 4.

b) Use a Boolean mask to access all element whose value is in the interval $[5, 10]$.

c) Use a Boolean mask to access all element whose value is in the interval $[2, 4]$ or $[6, 9]$.

Question 6

Take the array $x = [0, 1, 2, 3]$ and convert it to

$$\begin{bmatrix} 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \end{bmatrix}.$$

Question 7

Construct the following matrix M using appropriate NumPy functions starting from the array $[1, 2, 4]$:

$$\begin{bmatrix} 1 & 1 & 2 & 2 & 4 & 4 \\ 1 & 1 & 2 & 2 & 4 & 4 \end{bmatrix}.$$

Question 8

Write a function `blocks(m,n)` that, for given inputs n and m, returns an $(m+n) \times (m+n)$ matrix that contains an $m \times m$ block of ones on the top left, and an $n \times n$ block of ones on the bottom right (and zeros elsewhere). Use `hstack()` and `vstack()` in your solution.

For $m = 2$ and $n = 3$, this results in the matrix

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Question 9

Write a function `checkerboard(n)` that returns a checkerboard pattern of zeros and ones of size $n \times n$ (see examples below; the top-left element is always a 1).

For $n = 5$ and $n = 6$, the matrix should look like this

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Question 10

Compute the anti-diagonal of the matrix

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 5 & 6 & -7 \\ -9 & 10 & 11 \end{bmatrix}$$

using `np.rot90()`. The anti-diagonal is obtained by going from the bottom-left to the top-right element, i.e., $[-9, 6, 3]$ in this case (and not $[3, 6, -9]$). Have a look at the documentation of `np.rot90()` for details of how this function works.