Programming for Computational Linguistics 2025/2026

Exercise Set 4 (2025-11-13)
Lists

Once you have completed all exercises, send an email to the progclgrader@ims.uni-stuttgart.de with the title "submit set4", and with your python files as attachments. You should receive a response with feedback and a grade.

Exercises are due on **2025-12-02 23:59**, but we encourage you to do them during the lab session.

Exercise 1. Create a module `list_utils.py` with the following functions:

- `last(l)` – returns the last element of list `l`.
  For example, `last([1, 2, 5, 4])` should return `4`
- `middle(l)` – returns the middle element of list `l` with an odd number of elements.
  For example, `middle([1, 2, 4, 8, 16])` should return `4`
- `product(l)` – returns the product of a list of numbers.
  For example, `product([1, 2, 3, 4, 5])` should return `120`
- `mean(l)` – returns the mean (average) of a list of numbers.
  For example, `mean([1, 2, 3, 4, 6])` should return `3.2`
- `even_sum(l)` – returns the sum of the elements of a list with even indices.
  For example, `even_sum([2, 3, 5, 7, 11])` should return `18` $(2 + 5 + 11)$.

All functions should accept a list `l` as a parameter, and should not modify that list.

Exercise 2. Create a module `matrix_utils.py`. This module will deal with matrices – two-dimensional arrays of values. In python, we will represent a matrix as a list of lists of equal length. Within this module, write the function `diagonal(matrix)`. This function should accept one parameter `matrix`, which is a square matrix – a list of n sublists, where each sublist has n elements. Your function should return the diagonal of that matrix as a list. For example, the diagonal of the matrix $\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$ is $\begin{bmatrix}1 & 5 & 9\end{bmatrix}$. Thus, your function should behave as follows:

```python
A = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
print("A's Diagonal:", diagonal(A))
B = [[1,2,3,4],[12,13,14,5],[11,16,15,6],[10,9,8,7]]
print("B's Diagonal:", diagonal(B))
```

```
A's Diagonal: [1, 5, 9]
B's Diagonal: [1, 13, 15, 7]
```

Exercise 3. Within the module `matrix_utils.py` write a function `transpose()`. The function should accept one parameter `matrix` which is a two-dimensional matrix (in Python it is a list of lists of the same size). The function should return a new, transposed matrix. Example usages of this functions:

```python
matrix1 = [[0, 1, 2, 3, 4],
           [0, 1, 2, 3, 4] ]

print("Transposed:", transpose(matrix1))
```

```
Transposed: [[0, 0], [1, 1], [2, 2], [3, 3], [4, 4]]
```

```python
matrix2 = [['a', 'b'],
           ['c', 'd'] ]

print("Transposed:", transpose(matrix2))
```

```
Transposed: [['a', 'c'], ['b', 'd']]
```