Programming for Computational Linguistics 2025/2026

Exercise Set 3 (2025-11-06)
Functions

Once you have completed all exercises, send an email to the progclgrader@ims.uni-stuttgart.de with the title "submit set3", and with your python files as attachments. You should receive a response with feedback and a grade.

Exercises are due on **2025-11-25 23:59**, but we encourage you to do them during the lab session.

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

- `is_prime` with one integer parameter `n`. The function should return `True` if the number `n` is a prime number and `False` otherwise.
  Example usages of this function:
  - `is_prime(1)` returns False
  - `is_prime(2)` returns True
  - `is_prime(10)` returns False
  - `is_prime(-7)` returns False
  - `is_prime(104593)` returns True
- `is_twin_prime` with one integer parameter `n`. The function should return `True` if the number `n` is a twin prime number and `False` otherwise.
  - `is_twin_prime(10)` returns False
  - `is_twin_prime(37)` returns False
  - `is_twin_prime(421)` returns True
  - `is_twin_prime(1091)` returns True
  - `is_twin_prime(-13)` returns False

Exercise 2. Random passwords are safe but it is hard to remember them. Your task will be to write a function which will generate an easy to remember random password. Such password will consist of short fragments of text which are easy to pronounce. Function to sample such fragments is in file `fragments.py` (download it from ILIAS) and is called `get_random_text`. The function can return fragments only of length 2, 3, or 4. An example usage of this function looks like this:

```
from fragments import get_random_text

i = 0
while i < 3:
    print('Random text: ' + get_random_text())
    i += 1
```

```
Random text: an
Random text: num
Random text: sy
```

Create a module `passwords.py` with function `easy_password` (submit only this file, not the `fragments.py`). The function should accept one integer parameter `length` (assume that `length` > 1) and generate a password with exactly `length` characters. An example usage of this function:

- `easy_password(10)` can return 'getbacuson'
- `easy_password(10)` can return 'piaftromer'
- `easy_password(15)` can return 'spresedparmonset'
- `easy_password(2)` can return 'en'

Remember, that the password has to be built from **full fragments** given by `get_random_text`. Basically an approach, where you generate a password that is too long and then cut few last characters, is not correct.

Exercise 3. Create a python program `figure.py` with the code below:

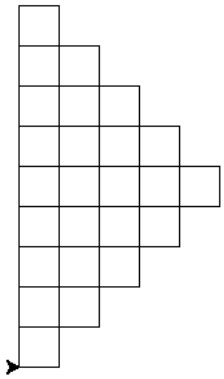
```
from turtle import * # imports turtle

def draw_square():
    i = 0
    while i < 4:
        forward(30)
        right(90)
        i += 1


draw_square()

# the purpose of this line is to keep
# the picture on the screen
input()
```

Run it and see what happens. Then, change the code so that it draws the following shape:



As this excercise will be graded manually, you will receive no autograder feedback. We will only grade your final submission, and you will receive your grade after the submission deadline.