Practice Exam Scientific Programming 2

Date: ...

Name:

.....

- This is a digital exam. The exam consists of 3 assignments in which you have to write a short python program.
- Create one file for all your solutions called `sp2_exam_practice.py`. This is the file you'll hand in at the end of your exam.
- This is a **closed-book exam**, meaning:
  - Close all programs on your laptop, except:
    - The Pulsar editor (with only the python file for this exam open)
    - Your terminal
    - The submit page for this exam: sp.proglab.nl/sp2exam/
    - The official Pandas documentation
  - You're not allowed to use any other webpage.
  - You're not allowed to look at any existing code you've written before this exam.
  - You cannot get any help with programming during the exam.
- You're only evaluated based on the *correctness* of your solutions, code design is not important. So, **you don't have to worry about comments or style guides**.
- You're only allowed to import the `pandas` module, but no other modules.
- You can test your code using checkpy. First download the tests for the exam:

  ```
  checkpy -d /spcourse/exam-tests
  ```

  Run checkpy:

  ```
  checkpy sp2_exam_practice
  ```

- Submit your solutions on the website when you're done. **Check with the teacher present if you handed in your assignment correctly before leaving the exam venue.**
- After submitting, hand in this exam paper showing your student card or ID.

Time submitted (to be filled out by teacher):

.....

1 After is

Write a Python function called `after_is(text)` that takes a text string as input and returns keeps track which word directly follows the word 'is' or 'was'. The function returns a dictionary where the keys are the words that directly occur after 'is' or 'was' and the values count how often they occur in that position..

The function should consider words case-insensitively and should ignore common punctuation marks such as commas, periods, parentheses, exclamation marks, and question marks. You can use the following `tokenize()` function to convert the text to a list of words:

```
def tokenize(text):
    return [word.strip(" ,\n.);(!?)'").lower() for word in text.split(' ')]
```

Example usage:

```
text = """It was a place where up is down, and down is up; where nothing is
quite what it seems. Why is a raven like a writing desk?
"""
print(after_is(text))
```

Expected output:

```
{'a': 2, 'down': 1, 'up': 1}
```

2 Translations

Imagine we have two python dictionaries for translations. For instance an English-French and a French-Spanish dictionary. Write a function `combine_dicts(dict1, dict2)`, that combines them to create an English-Spanish dictionary.

Example usage:

```
english_to_french = {'banana': 'banane', 'apple': 'pomme', 'almond': 'amande',
                     'cat': 'chat', 'fine': 'amande'}
french_to_spanish = {'pomme': 'manzana', 'car': 'coche', 'banane': 'plátano',
                      'amande': 'almendra'}

english_to_spanish = combine_dicts(english_to_french, french_to_spanish)
print(english_to_spanish)
```

Expected output:

```
{'banana': 'plátano', 'apple': 'manzana', 'almond': 'almendra', 'fine': 'almendra'}
```

3 Ingrid Bergman

The file `films.csv` contains data of over a 1000 films from before the 2000s. In this exercise, you will use `films.csv` to answer a question about the data. The contents of the file look as follows:

```
Year;Length;Title;Subject;Actor;Actress;Director;Popularity;Awards
1990;111.0;Tie Me Up! Tie Me Down!;Comedy;Banderas, Antonio;Abril, Victoria;Almodvar, Pedro;68.0;No
1991;113.0;High Heels;Comedy;Bos, Miguel;Abril, Victoria;Almodvar, Pedro;68.0;No
1983;104.0;Dead Zone, The;Horror;Walken, Christopher;Adams, Brooke;Cronenberg, David;79.0;No
...
1987;103.0;Heat;Mystery;Reynolds, Burt;Young, Karen;Jameson, Jerry;69.0;No
1947;87.0;Night Is My Future;Drama;Malmsten, Birger;Zetterling, Mai;Bergman, Ingmar;17.0;No
1990;92.0;Witches, The;Science Fiction;Fisher, Jasen;Zetterling, Mai;Roeg, Nicolas;18.0;No
```

The films have no particular order, and for each film the file contains the following information:

1. Year: the year in which the film was released
2. Length: the duration of the film in minutes
3. Title: the title of the film
4. Subject: the genre of the film
5. Actor: the name of the main actor in the movie
6. Actress: the name of the main actress in the movie
7. Director: the name of the director of the movie
8. Popularity: the popularity of the movie on a scale from 0 to 100
9. Awards: whether the movie has gotten an award (Yes/No)

The columns with names always only contain one name.

Write a function `most_frequent_actress(filename)` that finds the actress that has the most appearances in this data file. Have the function also return a list of all the movies the actress appears in. Using `pandas` for this assignment is optional.

Example usage:

```
actress, movies = most_frequent_actress('films.csv')
print(actress)
print(movies)
```

Expected output:

```
Bergman, Ingrid
['Count of Old Town, The', 'Autumn Sonata', 'Gaslight', 'Indiscreet', 'Walpurgis Night',
'Joan of Arc', 'A Woman Called Golda', 'A Walk in the Spring Rain', 'Under Capricorn',
'Notorious', 'June Night', 'Goodbye Again', 'Anastasia', "Bells of St. Mary's, The",
'Intermezzo', "A Woman's Face", 'Swedenhielms', 'Only One Night', 'Dollar', 'Elena & Her Men',
'Europa Fifty-One', 'Voyage in Italy', 'Fear', 'Stromboli', 'Cactus Flower']
```