Homework No. 04

Due: 23:59, 19 November, 2023

Max points: 100

Rules

- **No late homeworks.** A penalty of 10 points is applied for each day.
- **No plagiarism.** Collaboration is encouraged, but copying someone else's work without proper attribution is not admitted and invalidates the submission. A penalty is applied to all parties included.

Submission procedure

- Each problem solution should be saved in a separate file. The following naming convention should be used: `problem{number}.{extension}`. For example, `problem1.py` or `problem1.pdf`.
- At the start of each file, homework number, student full name and problem number should be mentioned. For example:

```
"""
Homework 4
Name: John Doe
Problem 1
"""
```

- Solution files should be uploaded to YSU Moodle. Alternatively, you can commit your solutions to a Git repository and provide the repository URL on Moodle.

Problem 1.1 [10 points]

Write a python function named `generate_rectangle` that takes the position of the top-left corner of a rectangle, its width and height as input and returns a list of points that form the rectangle.

Example

```
def generate_rectangle(x, y, width, height):
    pass

print(generate_rectangle(0, 0, 3, 2)) # [(0, 0), (1, 0), (2, 0), (0, 1),
(1, 1), (2, 1)]
```

Problem 1.2 [20 points]

Write a python function named `rectangle_iou` that takes two rectangles as input and returns the intersection over union (IoU) of the two rectangles. The rectangles are represented as a list of points that form the rectangle. The IoU is calculated as the number of points in the intersection of the two rectangles divided by the number of points in the union of the two rectangles.

$$\text{IoU} = \frac{\text{Number of points in the intersection of the two rectangles}}{\text{Number of points in the union of the two rectangles}}$$

Example

```
def rectangle_iou(rect1, rect2):
    pass

rect1 = [(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2)]
rect2 = [(1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2)]

print(rectangle_iou(rect1, rect2)) # 0.33
```

Problem 2.1 [20 points]

Write a Python function called `find_unique_words` that takes a string of text as input and returns a set of unique words found in that text. The function should treat words as case-insensitive and ignore punctuation marks.

- The function should be case-insensitive (i.e., "Hello", "hello", and "HELLO" are considered the same word).
- The function should ignore punctuation (i.e., "hello!", "hello", and "hello?" are considered the same word).

Example

```
def find_unique_words(text):
    pass

print(find_unique_words("Hello, world! Hello universe.")) # {'hello',
'world', 'universe'} (order may vary)
print(find_unique_words("To be, or not to be, that is the question.")) #
{'to', 'be', 'or', 'not', 'that', 'is', 'the', 'question'} (order may
vary)
```

Problem 2.2 [20 points]

For this task, you are required to write a Python function named `find_common_words` that compares two text samples and returns a set of words that are common to both samples. The function should treat words as case-insensitive and ignore punctuation marks.

Example

```
def find_common_words(text1, text2):
    pass

text1 = "Hello, world! Hello universe."
text2 = "Hello everyone, this is the world speaking."

print(find_common_words(text1, text2)) # {'hello', 'world'} (order may
vary)
```

Problem 2.3 [30 points]

Develop a Python function named `calculate_text_similarity_percentage` that calculates the similarity between two text samples based on their common words, excluding a given set of stop words. The similarity is defined as the percentage of common words (excluding stop words) in the texts, compared to the total number of words in both texts.

Requirements:

- Words should be considered in a case-insensitive manner.
- Punctuation marks should be ignored.
- Stop words provided in the stop_words set should be excluded from both texts before the calculation.
- The similarity percentage is calculated as: $\frac{\text{Number of common words (excluding stop words)}}{\text{Total number of words (excluding stop words) in both texts}} \times 100$
- The function should return the similarity percentage as a floating-point number rounded to 2 decimal places.

Example

```
def calculate_text_similarity_percentage(text1, text2, stop_words):
    pass

stop_words = {"the", "a", "an", "and", "or", "but", "is", "are", "am",
"was", "were", "be", "been", "being", "have", "has", "had", "do", "does",
"did", "shall", "will", "should", "would", "may", "might", "must", "can",
"could"}

text1 = "Once upon a time, in a land far away, there lived a wise old
king. The king ruled his kingdom with fairness and justice, earning the
love and respect of his people. Every day, the king would walk through the
kingdom, listening to the concerns of his subjects and offering wisdom and
guidance. His kingdom was known for its prosperity and peace, a beacon of
hope in a tumultuous world. The people of the land worked hard, but they
were happy, knowing that their ruler had their best interests at heart."

text2 = "In a distant land, long ago, there was a king known for his
wisdom. He governed his realm with equity and benevolence, which made him
beloved by his citizens. The monarch made it a point to roam his kingdom
regularly, paying attention to his people's needs and providing advice and
assistance. This kingdom was a place of wealth and tranquility, a symbol
of optimism in a troubled era. The inhabitants of this kingdom toiled
diligently, yet they felt content, secure in the knowledge that their
sovereign cared for their wellbeing."

print(calculate_text_similarity_percentage(text1, text2, stop_words)) #
15.00
```