Assignment 1

Problem 1 - Anagram

An anagram is a word or a phrase made by transposing the letters of another word or phrase; for example, "parliament" is an anagram of "partial men," and "software" is an anagram of "swear oft."

in file `Anagram.py`, implement a function `anagram(string1, string2)` that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation. Return `True` if and only if `string1` is an anagram of `string2`, return `False` otherwise.

Hint: You can use the following snippets to remove all the spaces and punctuations from a string:

```
In [1]: import re
   ...: string1 = "Data Structures rock."
   ...: regex = re.escape(" \\/.,()[]")
   ...: s1 = re.sub(f"[{regex}]", '', string1).lower()

In [2]: s1
Out[2]: 'datastructuresrock'
```

Problem 2 - Object-oriented programming

Implement class `Fraction` in file `Fraction.py`

Important:

- The coding canvas for this problem is provided in the assignment.
- Support the following operations:
  - Fraction + Fraction
  - Fraction += Fraction
  - Fraction – Fraction
  - Fraction * Fraction
  - Fraction == Fraction
  - print(Fraction)
- Your Fraction need *not* be the most simplified representation.
- You can define your own functions and call them. Just make sure the original provided test code runs without problem.
- Check your implementation's correctness with the given test code.

Problem 3 - Has duplicate

In file has_duplicate.py, implement function `has_duplicate(list1)` that determines whether `list1` contains duplicate values.

```
In [1]: has_duplicate([1, 3, 6, 2, 4])
Out[1]: False

In [2]: has_duplicate([1, 3, 6, 2, 4, 3])
Out[2]: True
```

Important:

- You can assume `list1` contains only integers.
- You can solve this problem in any way you like.
- What is the worst case runtime for your program? Mention your runtime as a comment in your `.py` file. (5pts)

Problem 4 - Buy two items

You receive a `credit` at a local store and would like to buy two items. You first walk through the store and create a `list1` of all available items' prices. From this `list1`, you would like to buy two items that add up to the entire value of the credit.

In file buy_two_items.py, implement function, `buy_two_items(credit, list1)` that returns a tuple of two integers. Those two integers came from `list1`, and should add up to `credit` exactly.

```
In [1]: buy_two_items(200, [150, 24, 79, 50, 88, 345, 3])
Out[1]: (150, 50)

In [2]: buy_two_items(295, [678, 227, 764, 37, 956, 982, 118, 212, 177, 597,
519, 968, 866, 121, 771, 343, 561])
Out[2]: (118, 177)
```

Important

- You can assume `list1` contains only integers.
- You can assume there always exists a pair in `list1` that will add up to `credit`.
- You can solve this problem in any way you like.
- What is the worst case runtime in your program? Mention your runtime as a comment in your `.py` file. (5pts)

Submission format

The files below constitute your coding canvas for this assignment.

1. `Anagram.py`
2. `Fraction.py`
3. `has_duplicate.py`
4. `buy_two_items.py`

You need to complete them, and then submit them directly (do not put them in a directory or zip) to gradescope:

https://www.gradescope.com/courses/399287/assignments/2080886/