MTH 3300 Problem Set 4

Due date: Mar 9, 2025 at 11:59PM

For the programming assignment, please follow the following naming convention for your Python files.
`mth3300_<lastname>_<firstname>_pset4_part<#>.py`

As with previous problem sets, the written portion should be a separate PDF.

Note that for this homework, I expect that all will provide fully-implemented functions that have at least passed the basic tests I've provided.

1. (20 points) Write a function called `rotate(items, k)` that rotates a list to the right by `k` places.

```
def rotate(items, k):
    ...

assert rotate_list([1, 2, 3, 4, 5], 2) == [4, 5, 1, 2, 3]
assert rotate_list(['a', 'b', 'c', 'd'], 1) == ['d', 'a', 'b', 'c']
```

2. (20 points) Write a function called `print_pyramid(n)` that returns a list of lists representing a number pyramid of height `n`. Each row should contain numbers increasing from 1 to the row number, then decreasing back to 1.

   Hint: This will require the use of nested loops

```
def print_pyramid(n):
    ...

expected = [
    [1],
    [1, 2, 1],
    [1, 2, 3, 2, 1]
]
assert print_pyramid(3) == expected
```

3. (20 points) Write a function called `lcp(words)` that returns the longest common prefix of a list of strings. If there is no string in common, then your function should return an empty string

```
def lcp(words):
    ...

assert lcp(["flower", "flow", "flight"]) == "fl"
assert lcp(["dog", "racecar", "car"]) == ""
```

4. (20 points) Write a function `has_duplicate(nums)` that returns `True` if there are any duplicate values within the list. Otherwise this returns `False`

   Hint: You'll want to use a `set` for this implementation

```
def has_duplicate(nums):
    ...

assert has_duplicate(["a", "ewrwe", "a", "b"]) == True
assert has_duplicate([1, 1, 2, 3]) == True
assert has_duplicate([1, 23, 4, 5]) == False
```