Python for Math and Stat Fall 2023
Exam 1

1. (20 pts) For the following 4 problems, write down what each code block would display if executed in a Jupyter cell.

   (a)
   ```
   (1703 // 170, 1703 % 10)
   ```

   (b)
   ```
   mascot = ’buffalo’
   2 * mascot[:2] + ’2’
   ```

   (c)
   ```
   [j > 2 for j in range(-2, 2)]
   ```

   (d)
   ```
   zz = 0
   for k in range(2, 5):
       print(zz)
       zz += (-1)**k * k
   ```

   Solution:

   (a) `(10, 3)`

   (b) `’bubu2’`

   (c) `[False, False, False, False]`

   (d)
   ```
   0
   2
   -1
   ```

2. (10 pts) Use a **list comprehension** to create the list of 20 numbers shown below.
   (*Hint:* Recall that `range` takes only integer arguments.)

   ```
   [100.0, 99.6, 99.2, 98.8, ...]
   ```

   Solution:

   ```
   [100 - 0.4*i for i in range(20)]
   ```

   OR

   ```
   [n / 10 for n in range(1000, 920, -4)]
   ```

3. (10 pts) Write a function `plurals(nouns)` that takes a list of nouns in string format and returns a new list containing the nouns in plural form, using these two rules:

   - If the noun ends in `s` or `ch`, add `es` to the end of the string.
   - Otherwise add `s` to the end of the string.

Assume that each element of `nouns` consists of at least 2 characters.

Example:
`plurals([’boat’, ’iris’, ’bench’])` returns `[’boats’, ’irises’, ’benches’]`.

Solution:

```
def plurals(nouns):
    result = []
    for n in nouns:
        if n[-1] == ’s’ or n[-2:] == ’ch’:
            result.append(n + ’es’)
        else:
            result.append(n + ’s’)
    return result
```

4. (10 pts) Write a function `middle(nums)` that takes a non-empty list of numbers and returns

    - the number in the middle position of the list if the list has an odd number of elements, or
    - the average of the two numbers in the middle positions if the list has an even number of elements.

    Assume that each element of `nums` is an `int` or `float`.

    Example:
    `middle([4, -3, 80, 6, 7.2])` returns `80`.
    `middle([4, -3, 80, 85, 6, 7.2])` returns `82.5`.

    Solution:

```
def middle(nums):
    numsct = len(nums)
    mid = numsct // 2

    if numsct % 2 == 1:
        return nums[mid]
    else:
        return (nums[mid-1] + nums[mid]) / 2
```