Problem 1. (*Sine Function*) Implement the function `sin()` in `sin.py` that calculates the sine of the argument x in radians, using the formula:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots .$$

Hint: In order to avoid the inaccuracies caused by computing with huge numbers, follow the approach described on page 97 of the IPP text for computing the function e^x.

```
$ python3 sin.py 60
0.8660254037844385
0.8660254037844386
```

Problem 2. (*Euclidean Distance*) Implement the function `distance()` in `distance.py` that returns the Euclidean distance between the vectors x and y represented as one-dimensional lists of floats. The Euclidean distance is calculated as the square root of the sums of the squares of the differences between the corresponding entries. You may assume that x and y have the same length.

```
$ python3 distance.py 5
-9 1 10 -1 1
-5 9 6 7 4
13.0
```

Problem 3. (*Palindrome*) Implement the function `is_palindrome()` in `palindrome.py` that returns `True` if the argument s is a palindrome (ie, reads the same forwards and backwards), and `False` otherwise. You may assume that s is all lower case and doesn't any whitespace characters.

```
$ python3 palindrome.py bolton
False
$ python3 palindrome.py amanaplanacanalpanama
True
```

Problem 4. (*Reverse*) Implement the function `reverse()` in `reverse.py` that reverses the one-dimensional list a in place, ie, without creating a new list.

```
$ python3 reverse.py
to be or not to be that is the question
<ctrl-d>
question the is that be to not or be to
```

Problem 5. (*Transpose*) Implement the function `transpose()` in `transpose.py` that creates and returns a new matrix that is the transpose of the matrix represented by the argument a. Note that a need not have the same number rows and columns. Recall that the transpose of an m-by-n matrix A is an n-by-m matrix B such that $B_{ij} = A_{ji}$, where $1 \leq i \leq n$ and $1 \leq j \leq m$.

```
$ python3 transpose.py
2 3
1 2 3
4 5 6
1.0 4.0
2.0 5.0
3.0 6.0
```

Files to Submit

1. `sin.py`
2. `distance.py`
3. `palindrome.py`

4. `reverse.py`
5. `transpose.py`

Before you submit:

- Make sure your programs meet the input and output specifications by running the following command on the terminal:

  ```
  $ python3 run_tests.py -v [<problems>]
  ```

  where the optional argument `<problems>` lists the problems (`Problem1`, `Problem2`, etc.) you want to test, separated by spaces; all the problems are tested if no argument is given.

- Make sure your programs meet the style requirements by running the following command on the terminal:

  ```
  $ pycodestyle <program>
  ```

  where `<program>` is the `.py` file whose style you want to check.