Programming Homework #1

** Please show all your answers within or as a result of cell execution in Jupyter notebook.*

e.g.,

Question 1-(1) Test with single input array

```
In [3]: A = [23, 12, 56, -1, 23, 32, 3, 0, 10, 16]
        insertionSort(A)
        print(A)

        [-1, 0, 3, 10, 12, 16, 23, 23, 32, 56]
```

Q1. Let's consider a sorting problem as follow:

Input: A sequence of *n* numbers $\langle a_1, a_2, \cdots, a_n \rangle$

Output: A reordering $\langle a'_1, a'_2, \cdots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

(1) Solve this problem by implementing **insertion sort.** Show your answer when the input A is [23, 12, 56, -1, 23, 32, 3, 0, 10, 16].

<20 pts>

(2) We will compare running times of insertion sort for three different types of input, e.g., *random, sorted(best-case), reverse-sorted(worst-case) input*, with the following input sizes (n):

n = [100, 1000, 2000, 3000, 4000, 5000]

i.e., each type of input consists of six different sizes of input. (See the **Generate inputs** cell of *template.ipynb* file as a reference)
Then, we will measure the running time for all the input set and compare them by plotting as a graph. (See the **Test and plot** cell of *template.ipynb* file as a reference)
Plot the results as a result of cell execution. (*You can reuse codes in *template.ipynb* file)

<10 pts>

(3) Based on the result of (2), explain the trend of running time for different types of input. You can explain the trend by using the asymptotic notation in the cell directly using Markdown syntax (See the **Example to write equations with Markdown** of *template.ipynb* file as a reference)

<10 pts>

Q2. Merge sort

(1) Show the answer of Q1-(1) by implementing **Merge-sort**.

<20 pts>

(2) Generate *random input* with the following input sizes (n):

n = [100, 1000, 2000, 3000, 4000, 5000]

Compare the running time of ***Insertion-sort*** *and* ***Merge-sort*** by plotting a graph as Q1-(2).

(*You may need to regenerate random input and copy it into another parameter. Because once we run a sort algorithm, that input will be sorted, so we cannot run another algorithm with the same input.)

<10 pts>

(3) Based on the Q2-(2) result, compare the running time of the Insertion sort and Merge sort. You can explain the trend by using the asymptotic notation in the cell directly using Markdown syntax.

<10 pts>

Q3. Implement **Bubble sort** with the following pseudocode. Show the result by using randomly generated generating random input with the size of 20.

```
BUBBLESORT(A, n)
1  for i = 1 to n − 1
2      for j = n downto i + 1
3          if A[j] < A[j − 1]
4              exchange A[j] with A[j − 1]
```

<10 pts>

Q4. Implement with the following pseudocode. Show the result by generating random input with the size of 20.

```
SELECTION-SORT(A, n)
for i = 1 to n − 1
    smallest = i
    for j = i + 1 to n
        if A[j] < A[smallest]
            smallest = j
    exchange A[i] with A[smallest]
```

<10 pts>