Exercise 12. Answer Sheet

Student's Name: Yuta Nemoto Student's ID: s1240234

Problem 1. (40 points) Implement the randomized algorithm for calculating π given in the lecture.
- Calculate π using your program 10 times using N number of points from the next table. Fill in the results you got.

N	1	2	3	4	5	6	7	8	9	10
1	4.0	4.0	0.0	0.0	4.0	4.0	4.0	4.0	4.0	4.0
10	2.8	3.2	3.6	2.8	2.8	3.2	3.6	3.2	4.0	3.2
100	2.92	3.24	3.24	3.12	2.92	3.12	3.08	3.08	3.04	3.28
1000	3.064	3.076	3.144	3.056	3.108	3.088	3.112	3.048	3.036	3.092
10000	3.128	3.1532	3.1468	3.158	3.1508	3.146	3.1252	3.1664	3.1244	3.116
100000	3.1382	3.14296	3.14416	3.14768	3.14108	3.14404	3.14212	3.14288	3.1364	3.14988

- Upload your code.

<How to compile/run>
Command:
javac RandomizedAlgorithms.java
java RandomizedAlgorithms
Then it automatically output the list above like the screenshot below.

```
std6dc12{s1240234}53: javac RandomizedAlgorithms.java
[1]  + Done                       emacs RandomizedAlgorithms.java
std6dc12{s1240234}54: java RandomizedAlgorithms
N       1       2       3       4       5       6       7       8       9       10
1       4.0     4.0     0.0     0.0     4.0     4.0     4.0     4.0     4.0     4.0
10      2.8     3.2     3.6     2.8     2.8     3.2     3.6     3.2     4.0     3.2
100     2.92    3.24    3.24    3.12    2.92    3.12    3.08    3.08    3.04    3.28
1000    3.064   3.076   3.144   3.056   3.108   3.088   3.112   3.048   3.036   3.092
10000   3.128   3.1532  3.1468  3.158   3.1508  3.146   3.1252  3.1664  3.1244  3.116
100000  3.1382  3.14296 3.14416 3.14768 3.14108 3.14404 3.14212 3.14288 3.1364  3.14988
std6dc12{s1240234}55:
```

Problem 2. (60 points) Write a program implementing the quicksort algorithm. Make two versions:
a) Randomized quicksort, where the pivot element is chosen at random.
b) Deterministic quicksort, where the pivot element is always the first element of the array.

- Generate random sequence of length N (as given in the Table)
- Measure the time each quicksort version needs to sort the sequences (fill in the average of 100 runs).

N	100	1000	10000	100000	1000000
Randomized	3[ms]	9[ms]	41[ms]	344[ms]	3217[ms]
Deterministic	2[ms]	7[ms]	39[ms]	333[ms]	3101[ms]

- Upload your code.

<How to compile/run>
Command:
javac QuickSortAlgorithms.java
java QuickSortAlgorithms
Then it automatically output the list above like the screenshot below.
(You can change the value of N by changing a constant value in the head of program.)

```
Original Array: 48 94 43 64 49 23 44 82 68 70 44 52 31 66 11 26 17 79 18 90 53 33 38 52 63 49 8 13 58 31 27 42 16 58 27 49 5
Randomized Quick Sort: 1 2 2 3 5 7 7 8 11 11 13 15 16 16 16 16 17 17 17 18 18 22 23 25 26 27 27 27 27 27 28 29 31 31 31 3
Deterministic Quick Sort: 1 2 2 3 5 7 7 8 11 11 13 15 16 16 16 16 17 17 17 18 18 22 23 25 26 27 27 27 27 27 28 29 31 31 31 3
Time each quicksort version needs to sort the sequences(value of N: 100):
Randomized: 3[ms]
Deterministic: 2[ms]
```