CS2124 Data Structures
Assignment 1: Function Runtimes Table

Completing the Program (15 points)

This program prints a table of runtimes (these are displayed in seconds) for given functions on arrays.

The program tests different array sizes to establish a relationship between input size and runtime. It tests each array size multiple times and then takes an average of the times. We also output how much the average runtime increased relative to the previous average. This is calculated by dividing the current average by the previous average (output "N/A" for the first increase value).

Here are example calls to the timing functions:

```
int sizes1[] = { 500, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000};

fRT = timeAlgorithm("Insertion Sort", 6, 6, sizes1, insertionSort );
printRuntimeTable(fRT);
freeFunctionRuntimes(fRT);

fRT = timeAlgorithm("quicksort", 12, 9, sizes1, quickSort );
printRuntimeTable(fRT);
freeFunctionRuntimes(fRT);
```

This results in following table:

```
This solution was completed by:
<student name>

Insertion Sort
Test size Test #1    Test #2    Test #3    Test #4    Test #5    Test #6    Average   Increase
500       0.000      0.001      0.001      0.001      0.000      0.001      0.001     N/A
1000      0.002      0.002      0.002      0.002      0.002      0.002      0.002     3.717
2000      0.007      0.007      0.007      0.007      0.007      0.007      0.007     3.833
4000      0.028      0.028      0.028      0.028      0.028      0.028      0.028     3.911
8000      0.110      0.109      0.111      0.111      0.111      0.112      0.110     3.933
16000     0.442      0.442      0.442      0.440      0.444      0.437      0.441     3.998

quicksort
Test size Test #1    Test #2    Test #3    Test #4    Test #5    Test #6    Test #7    Test #8    Test #9    Test #10   Test #11   Test #12   Average   Increase
500       0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000     N/A
1000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000     1.985
2000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000     2.131
4000      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001      0.001     2.054
8000      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002      0.002     2.085
16000     0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004      0.004     2.148
32000     0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008      0.008     2.098
64000     0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016      0.016     2.063
128000    0.035      0.034      0.034      0.034      0.034      0.034      0.034      0.034      0.034      0.034      0.034      0.033      0.034     2.101
```

Note your runtimes will vary a bit since the test data is randomly generated.

The runtimes are stored in a functionRuntimes struct. You are completing a program to create and fill data in this struct, print the data of this struct, and free this struct.

You are given a partial implementation in the file "runtimeTable.c". Specifically you are tasked to complete the functions below the heading "Functions for finding and printing runtime" (there will also be a "TODO" next to anything you need to change/complete). No other functions should be changed.

Using the Program (5 points)

After you have the program completed, you should use it to help determine the asymptotic runtimes of the three mystery functions (i.e., mysteryRuntime1, mysteryRuntime2, mysteryRuntime3).

Be sure to also examine the code of the mystery functions to confirm/improve your estimations.

Fill in the following table in the provided C file:

```
/*
TODO: Give your asymptotic estimates for the runtimes of the following 3 functions:

mysteryRuntime1:  O(   )
mysteryRuntime2:  O(   )
mysteryRuntime3:  O(   )
*/
```

Deliverables:

Your solution should be submitted as "runtimeTable.c". Be sure to fill in the runtimes described above as well as fill in your name in the "runtimeTable.c" file.

Upload this file to Blackboard under Assignment 1. **Do not zip your file**.

To receive full credit, your code must compile and execute. You should use valgrind to ensure that you do not have any memory leaks.