M1: Lab Worksheet

Part I

Compute the exact number of times that the statement TICK is executed in the following pseudocode fragments, as a function of n. **Show your work**, including for each loop the minimum and maximum value of the loop counter as a function of n and/or the next outermost counter. Give your answer in closed form, with no remaining summations. Work must be shown to receive full credit.

Reminder: what is "pseudocode"? It is a way of expressing algorithms that describes at a high level what they do but omits details specific to any particular language (Java, C, Python, etc.). You'll find lots of pseudocode in your textbook. If you have any question about what the code blocks below mean, please ask.

The main thing that may be new to you in pseudocode is the use of ← to indicate assignment: "$j \leftarrow 1$" would be written in Java as "`j = 1;`". Pseudocode uses ← because it often uses "=" to mean equality testing (where Java would use `==`).

1.
```
for $j$ in $1 \ldots n$ do
    for $k$ in $0 \ldots j+5$ do
        TICK
```

2.
```
$i \leftarrow 0$
while $i < n$ do
    TICK
    $j \leftarrow n$
    while $j > i$ do
        TICK
        TICK
        $j--$
    $i++$
```

3.
```
TICK
$i \leftarrow 1$
while $i \leq n$ do
    TICK
    $j \leftarrow 0$
    while $j < i \times i$ do
        TICK
        $j++$
    $i++$
```

Part II

Answer each of the following questions. **Justify your answers** using either the definitions of O, Ω, and Θ or the techniques shown in class (along with basic math).
Note: for limit tests involving logs, you may ignore any constants generated by taking derivatives of the log; that is, you may assume its base is e. This assumption is permissible because any log can be converted to a natural log by multiplying by a constant, which does not affect the outcome of the test.

4. Does $5 \cdot (n+2)^2 = \Omega(n \log n)$?

5. Does $8^{\log_2 n + \log_2 \log_2 n} = \Omega(n^3)$?

6. Does $n \log_{16} n = O(n \ln n)$?

7. Does $(3n^2 - 10n) = \Theta(n^2)$?

8. Does $n \log n = \Omega(n^{11/8})$?

9. Let $f(n)$ and $g(n)$ be non-negative functions of n. Prove using the definitions of O, Ω, and Θ — not using limit tests — that if $g(n) = \Omega(f(n))$, then $f(n) + g(n) = \Theta(g(n))$.

10. Let $f(n)$ and $g(n)$ be non-negative functions of n.
If $f(n) = \Theta(g(n))$, does $f(n)/g(n) = \Theta(1)$? Justify your answer.

Part III

Every night, AT&T has to update its database of every customer's telephone number. To enable fast lookups, the database includes an index that is a sorted list of all its phone numbers. This index must be rebuilt each night by re-sorting the contents of the database.

AT&T has hired you as a consultant to analyze their computing needs for the nightly sorting run. After carefully checking GitHub, you have found three sorting algorithms – A, B, and C – that can sort n phone numbers in $4 \times 10^{-6} n^2$, $10^{-4} n \log_2 n$, and $1.4 \times 10^{-4} n$ seconds respectively on a single processor.

11. What is the smallest problem size n_0 such that algorithm B is strictly faster than algorithm A for all $n \geq n_0$? (*Hint:* Plugging in values for n or using Newton's method are acceptable ways to solve the problem, and you are welcome to use online tools such as Wolfram Alpha and Desmos.) Explain your reasoning.

12. What is the smallest problem size n_1 for which algorithm C is strictly faster than algorithm B for all $n \geq n_1$? Explain your reasoning.

13. Describe how to construct a sorting algorithm that always achieves the best running time of any of algorithms A, B, or C for a given n.

14. Suppose the phone database contains 10^8 phone numbers. The updated information arrives at 4 AM, and the company demands that the new database be ready for use by 5 AM, i.e. one hour later. To meet this deadline, you may split the database evenly across k processors and sort each part separately. (For this problem, ignore the cost of putting the pieces back together.) How many processors do you need to meet your deadline with each of algorithms A, B, and C?