Data Structures **Name:** _____
Prof. Bolton **Net ID:** _____
Assignment 1

This assignment contains 1 pages (including this cover page) and 4 questions. Total of points is 100.

Conditions: All work must be completed individually. No outside resources are permitted. The only permitted resources are your texts and class notes.

Include your name and Net ID. Follow submission instructions as indicated on Canvas.

---

1. (40 points) Complete the following Exercises from Drozdek 4th ed: 2.11.1.a, 2.11.2.a, 2.11.2.b, 2.11.2.d, 2.11.7, 2.11.8, 2.11.10.a - d, 2.11.11.

2. (20 points) Provide C++ code for an "in-place" **selectionSort** algorithm. Clearly define the domain of the input(s) and output and any constraints or assumptions. Explain why your implementation is "in-place". Describe (No formal proof necessary) the best-case and worst-case and the resulting Time and Space complexities (for both cases) in terms of the *size* of the input, i.e. the length of the list.

3. (20 points) Formally define a **factorial** algorithm which computes the factorial of its input.
    A. Clearly define the domain of the input(s) and output. And define the algorithm itself recursively (clearly identify the recurrence and initial conditions.)
    B. Prove time complexity of this function in terms of the input *value*. Define a step count recurrence (based on the recursive definition). For the worst-case, upperbound this recurrence (Big-O notation) using substitution or iteration methods discussed in class.

4. (20 points) Provide C++ code or pseudocode for the following Fibonacci algorithms, **Fib**($i \in \mathbb{N}$), which return the ith value in the Fibonacci sequence: 1,1,2,3,5,8,... . Clearly define the domain of the input(s) and output. Provide Big-O analysis for Time Complexity in terms of the *value* of the input. Justify your answers with a proof.
    A. Iterative **Fib** , Use Explicit Counting to define step count.
    B. Recursive **Fib** , Use Recurrence to define step count and Substitution to bound the step count. *Hint: Guess an exponential upperbound.*