SER 222 **Practice** Exam 1

Updated 9/2/2019

Last Name: ______________________

First Name: ______________________

Last 4 digits of ASU ID: ______________

Exam Instructions

The exam is open textbook (Algorithms 4e by Sedgewick and Wayne), as well as open note. No electronic items are allowed. Write legibly. Please use a pen (instead of a pencil) if you have one. There are 122 points available and the exam must be completed in 37.5 minutes. This exam has three types of questions:

Multiple choice questions: There are 0 points of multiple choice questions. An answer is selecting one option among the choices given. Each multiple choice is worth 2 to 5 points.

Short answer questions: There are 90 points of short answer questions. A typical answer is one or two sentences. Each short answer question is worth 5 or 10 points.

Programing questions: The programming questions are given near the end of the paper. They must be answered on the question paper. There are 32 points of write-in programming questions.

Topic	**Earned**	**Possible**
SA: Recursion		20
SA: Data Abstraction		20
SA: Stacks, Lists, and Generics		30
SA: Analysis of Algorithms		20
Prog: Recursion		24
Prog: Analysis of Algorithms		8
Total:		122

Short Answer: Recursion

1. Consider an algorithm for determining the size of some nested folders/files recursively. (Imagine right clicking an item on your desktop and viewing it's size.) What would be the base case(s) for this problem? [10 points]

2. Consider the problem of reversing letters of a string. Would it be more appropriate to use recursion or iteration? Justify your answer. [10 points]

Short Answer: Data Abstraction

3. When it comes to functionality (not performance), is it important to know how an ADT is internally implemented? Explain. [10 points]

4. Consider the following constructor for an immutable matrix ADT:

```
public class SolnMatrix implements Matrix {
    private final int[][] data;
    public SolnMatrix(int[][] matrix) {
        data = matrix
    }
    // the usual operations follow...
```

Will this class behave as expected? Explain. [10 points]

Short Answer: Stacks, Lists, and Generics

5. Assume that *head* is the head node of a singly linked list containing the nodes A, B, and C. What would the result of executing the following code be? Draw the resulting list using box and arrow notation and include any variables. [10 points]

```
head.getNext().getNext().setNext(head.getNext());
```

6. What would be the difference in memory usage for storing a thousand elements in an array vs a linked list? Which takes less space? Explain. [10 points]

7. If you were optimizing for performance and wanted to support potentially adding many new elements to a data structure, would an array or linked list be more appropriate? Explain using Big-Oh. [10 points]

Short Answer: Analysis of Algorithms

8. Consider the following growth function: [10 points]

   $f_1(n) = 100 + 10log_{10}(n)n^2 + 45n$

   What is the Big-Oh order of this function? You should provide a relatively tight upper bound.

9. What is the Big-Oh order of the following code fragment? The fragment is parametrized on the variable n. Assume that you are measuring the number of println calls. [10 points]

```
for (int i = 1; i <= n; i++)
  for (int j = 1; j <= n; j *= 10)
    System.out.println("Nested loops!");
```

N/A Programming: Recursion

10. Implement a recursive method called **countOccurrences** that takes the first node in a singly linked list and returns the number of times a particular value occurs in the list. For example, if countOccurrences("A") is called on a list with nodes containing: "A", "B", "C", "B", "E", it would return 1. If countOccurrences("B") was called on the same list, it would return 2. For reference, a standard implementation for the nodes of a singly linked list is given below.

```
public class LinearNode<T> {
  private LinearNode<T> next;
  private T element;
  public LinearNode(T elem) { next = null; element = elem; }
  public LinearNode<T> getNext() { return next; }
  public void setNext(LinearNode<T> node) { next = node; }
  public T getElement() { return element; }
  public void setElement(T elem) { element = elem; }
}
```

(a) Using the fantastic four approach, determine the size n problem for the method countOccurrences. [2 point]

(b) Identify the stopping condition(s) and the return value, if any, for the problem. [2 point]

(c) Determine the size m problem(i.e. the "sub-problem") for the problem. [2 point]

(d) How is the size-n problem constructed from the size m problem? [2 point]

(e) Implement the recursive method **public static int countOccurrences<T>(LinearNode node, T target)** (Hint: use .equals to compare the target with the contents of each node.) [16 points]

Short Answer: Stacks, Lists, and Generics

If this was the real exam, there might be a question here asking you to implement methods (potentially stand-alone, or in an ADT) using linked lists.

Programming: Analysis of Algorithms

11. Consider the following method, excerpted from a protein structural prediction algorithm. Assume that any variables not given as parameters are available as globals.

```
//sets initial interaction energy.
//  int n: dimension of square matrix storing protein backbone.
//  double[][] pair: energy matrix.
void energy() {
    double ee = 0;
    //reset all pair interaction energies to zero.
    for(int j = 1; j < n; j++)
        for(int i = 3; i < n-2; i++)
            pair[i][j] = 0.0;

    //<more code follows in actual program>
    //...
```

Give a growth function for *md_fragment* that counts the number of assignments in the inner loop based on n. [8 points]