Fall 2026 COP 3503H Exam #1
Date: 10/8/2026

Name: _____

1) (12 pts) Write a function that takes in an ArrayList of String, not necessarily distinct, and returns a HashMap<String,Integer> that stores a mapping between each **unique string** in the array list and a distinct integer identifier starting at 0. In addition, the identifiers should be assigned in alphabetical order. So if the original list stored ["pear", "orange", "pear", "apple", "strawberry"] then the map should be: "apple" → 0, "orange" → 1, "pear" → 2, "strawberry" → 3.

```
public static HashMap<String,Integer> makeMap(ArrayList<String> list) {

}
```

2) (8 pts) What is the minimum number of comparisons that are necessary to sort 6 items using a comparison sort? (Note: The answer is a positive integer.)

_____

3) (12 pts) Use the Huffman Coding algorithm to build a Huffman tree for the data below and then compute the number of bits the Huffman code saves. Assume the original encoding uses 3 bits per character:

Letter/Character	Frequency	Assigned Bit Code
A	12	
B	88	
C	13	
D	19	
E	16	
F	14	
G	18	
H	20	

Draw the process of building the tree, showing the substeps that are necessary for you to build the correct final tree and put a box around the final tree below. After building the tree, put the corresponding bit code for each letter in the last column of the chart above.

Bits Saved = _____

4) (12 pts) Determine the following sum, in terms of x, assuming that |x| < 1:

$$1 + 3x + 5x^2 + 7x^3 + 9x^4 + ...$$

(Note: Formally, the sum to find is $\sum_{i=0}^{\infty}(2i + 1)x^i$.) Put a box around your final result.

5) (9 pts) Trace through a Radix sort of the numbers below showing the intermediate results:

Original Numbers	First Iteration	Second Iteration	Third Iteration	Final Result
1234				1234
2298				1378
3256				1615
1615				1671
2136				2136
2324				2298
1671				2324
2655				2655
1378				3256

6) (16 pts) You're attempting to break a lock code. The code is ***n*** letters long. You know that the code has all distinct letters and you have a reference string, ***s***, where you know that each letter in ***s*** is incorrect, but each letter is within 3 of the correct letter in the code. For example, if the reference string = "SPAN", then you know that the first letter of the code has to be P, Q, R, T, U or V. Note that the third letter must be B, C or D since there are no letters before A. Complete the method below so that the initial method call (also shown below) prints out all possible strings that could be the correct code for the lock based on this information. Below a description of each formal parameter is given as well:

Method call to print all possibilities, assuming that s is correctly filled:

```
printPossible(new char[s.length], 0, s, new boolean[s.length]);

// curString - current String being built. k - how many letters
// have already been filled in for curString, s - reference string
// used is an array of size 26 where used[i] is set to true if
// letter i has already been filled in curString (one of the first
// k letters). When i=0, that's 'A', i=25 is 'Z'.
public static void printPossible(char[] curString, int k,
              char[] s, boolean[] used)

    if (k == s.length) {
        System.out.println(new String(curString));
        return;
    }
    // Fill in code below.









}
```

7) (15 pts) Given an integer array, **a**, of size ***n***, consider the problem of finding the number of ordered triplets (i, j, k) where i < j < k and a[i] < a[j] < a[k]. You may assume that all the values in the array are in the range [**1**, $\mathbf{10^6}$]. The goal is to solve the problem in O(nlgn) time. A binary index tree can help in this task. Assume that you have a class that already implements a binary index tree with the following methods:

```
// Creates a new binary index tree with indexes 1 though n,
// inclusive with all elements initially set to 0.
public bit(int n)

// Adds value to the index idx in this binary index tree.
public void add(int idx, long value)

// Returns the sum of the values in this binary index tree in
// all indexes from index 1 through index idx, inclusive.
public long query(int idx)
```

Complete the method below that takes in an integer array ***a*** and returns the desired number of triplets in O(nlgn) time. **(Note: This is a challenging question. For 12 pts partial credit, count the number of pairs i < j where a[i] < a[j] in O(nlgn) time. My solution uses 3 bit objects.)**

```
public static long countTriplets(int[] a) {




}
```

8) (15 pts) Design a greedy algorithm to solve the following problem (no code). Be as precise as possible and intuitively argue for the correctness of your algorithm:

There are **n + m** children, each of which will receive one candy for Halloween (sorry, times are tough!) There are **n** Nestle Crunch bars and **m** Milky Way bars to distribute to the children. Each child has various preferences for these two candies. Specifically, child i gets $\mathbf{a_i}$ units of happiness from a Nestle Crunch bar and $\mathbf{b_i}$ units of happiness from a Milky Way bar. Naturally, you want to distribute the candy such that the total sum of the children's happiness is maximized. Determine an algorithm that determines this maximum sum of happiness (and one way of distributing the candy to achieve this level of happiness.)

9) (1 pt) The book 438 Days is about a true story of Salvador Alvarenga's survival at sea on a small boat. How many days was he on his boat before he landed on a remote Pacific island?

_____

Scratch Page – Please clearly mark any work you would like graded.