COP 3503 Spring 2022 Section 1 Final Exam

Last Name: _____ First Name: _____

Date: 4/28/2022

1) (10 pts) Show the order of edges added into the minimum spanning tree of the graph with the adjacency matrix shown below when running Prim's algorithm starting from vertex **C**. Make sure to also indicate edges that were considered but not added because they would have created a cycle with the previously added edges.

From/To	A	B	C	D	E	F
A	--	8	--	--	9	10
B	8	--	--	2	7	5
C	--	--	--	10	8	--
D	--	2	10	--	3	3
E	9	7	8	3	--	4
F	10	5	--	3	4	--

Please fill out the table below in the order that the edges get pulled out from the priority queue in consideration for being added. Just write both letters denoting the edge. For example, the edge connecting B and D should be written as BD. If the edge gets added to the MST, write "Yes" in the second column. Otherwise, write, "No". It's possible there are more rows than necessary on the chart below.

Edge to Consider	Added? (Yes/No)

2) (6 pts) Consider running Karatsuba's algorithm for multiplying two eight-bit integers, X and Y, shown below, by breaking both down into two four-bit parts. The algorithm entails computing three four-bit multiplications, one of which could have a tiny bit of overflow. For the following example, show the three recursive multiplications that would be calculated using Karatsuba's:

```
X = 01011101
Y = 10110110
```

In each of the slots below, write **in binary** the two numbers that would have to be multiplied in the three recursive calls. Product #1 is the one that may end up having 5-bit number(s) in it. (Hint: The only actual computation that needs to be done in this problem is a tiny bit of binary addition.)

Product #1: _____ x _____

Product #2: _____ x _____

Product #3: _____ x _____

3) (7 pts) Consider counting the number of ways to make change for 14 cents using 1 cent, 3 cent, 5 cent and 8 cent coins. Fill in the table below showing the work that the dynamic programming algorithm to solve the problem would do to solve the problem. The first row has been filled out for you for convenience.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3														
5														
8														

4) (5 pts) What is the fewest number of comparisons necessary to sort 11 numbers via a comparison sort? Note: 11! = 39,916,800 and 2^{23} = 8,388,608.

_____

5) (9 pts) Show the result of running the dynamic programming algorithm to find the edit distance between the strings "ACGATTACGA" and "CTAGACTTGA" in the table below. The first row has been filled in for you.

	C	T	A	G	A	C	T	T	G	A
A	1	2	2	3	4	5	6	7	8	9
C										
G										
A										
T										
T										
A										
C										
G										
A										

6) (6 pts) The array below is the completed path array from running Floyd Warshall's Algorithm on a graph with vertices labeled 0 through 6. Using this array, write down the requested shortest paths. To write down a shortest path from vertex a to vertex b, write a sequence of 2 or more vertices, starting with a, ending with b, where each pair of consecutive vertices represents an edge on the shortest path between a and b. (Note: the array headers (indexes) are in bold and the contents of the array are not.)

From/To	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	2	3	4	0	2	5
1	6	1	3	1	0	2	5
2	6	2	2	1	0	2	5
3	6	2	3	3	0	2	5
4	6	2	3	4	4	2	5
5	6	2	3	6	0	5	5
6	6	2	3	6	0	2	6

Note: You may not use all the slots provided.

Shortest Path from 0 → 6: ____ , ____ , ____ , ____ , ____ , ____ , ____

Shortest Path from 5 → 4: ____ , ____ , ____ , ____ , ____ , ____ , ____

Shortest Path from 1 → 0: ____ , ____ , ____ , ____ , ____ , ____ , ____

7) (8 pts) Consider the following problem:

Given a string ***s*** and another string (text) ***t,*** determine the length of the longest prefix of ***s*** which is ALSO a substring of ***t.*** For example, if ***s*** = `racecar`, and ***t*** = `whereweracechance`, then the answer to the problem is 5, since the substring "`racec`" is contained in ***t***, but the substring "`raceca`" is not. Let ***m*** be the length of the string ***s*** and let ***n*** be the length of the string ***t.***

In class on April 19th (the second to last day), we learned of a probabilistic technique to find if one string is a substring of another string. Using this technique with an adaptation, in words, describe a detailed algorithm to solve this problem in O(***nlgm***) expected time. Cleary describe your algorithm **and explain why the expected run-time is O(*nlgm*).**

8) (10 pts) Consider the following game: You start with ***n*** marbles, where ***n*** is a positive integer. The game ends when you have no marbles left. In a single turn, a random number of marbles in between 0 and ***n,*** inclusive, are taken from you. (Each of these ***n***+1 choices is equally likely to occur.) You continue taking turns until you have no marbles left. Let T(***n***) represent the expected number of turns the game should take if you start with ***n*** marbles.

(a) (3 pts) Write down a recurrence relation that T(n) satisfies. (Note: your answer should have a summation in it.) In writing this down, do not do any simplification; just use the definition about how a single turn works in the game. (The initial condition/base case is T(0) = 0.)

(b) (7 pts) The equation from part (a) has the term T(n) appearing on both the left and right hand sides of the equation. These two terms can be combined so that T(n) appears on the left hand side only with terms ranging from T(0) to T(n-1) appearing on the right hand side. Also, noticing that T(0) = 0, T(0) can be removed. Do the algebra so that you have an expression for T(n) in terms of terms of the form T(i), where $1 \leq i \leq n-1$. (Note: when you do this, the form of your answer should be $T(n) = \frac{a}{b} + \frac{c}{d}\sum_{i=1}^{n-1} T(i)$, where a, b, c and d are relatively simple expressions, potentially in terms of ***n***.)

9) (10 pts) Write a static method in Java, **<u>using the iterative dynamic programming method</u>,** that takes in a non-negative integer, ***n***, and returns the value of T(n) from question 8. (Note: Yes, I am well aware that a correct answer on #8 is necessary to have a chance to solve this problem correctly!!! Also, as a hint, the run-time of your code should be $O(n^2)$.)

```
public static double t(int n) {




}
```

10) (6 pts) Write another static method that simulates one marble game. This method also takes in a non-negative integer ***n,*** simulates the game via a random object (assume a static variable belongs to the class that is shown below), and returns the number of turns the game took for the simulation. Note: Assume that r has already been instantiated in main before playGame has been called.

```
public static Random r;

public static int playGame(int n) {




}
```

11) (6 pts) Using the Master Theorem, write down the solution to the three recurrence relations shown below. (You should write down T(n) = O(?), where ? is a some function of n.)

(a) $T(n) = 4T\left(\frac{n}{2}\right) + O(n)$ _____

(b) $T(n) = 3T\left(\frac{n}{2}\right) + O(n^2)$ _____

(c) $T(n) = 64T\left(\frac{n}{4}\right) + O(n^3)$ _____

12) (6 pts) A common tactic to solve some type of problems is a technique called "Coordinate Compression". In this technique, we start with a set of input being a set of unique integers that aren't consecutive. (For example, the set could be {5, 2, 9, 16, 3, 12}.) For each number in the set, we would like to know its 0-based rank. When we sort the set, the order of the values is 2, 3, 5, 9, 12 and 16. The information we would like to be able to retrieve easily is the rank of any number in the list. For example, the rank of 2 is 0 and the rank of 12 is 4. To be able to retrieve this information, we would like to create a HashMap, mapping each number to its rank. In this example, the map would store 2 → 0, 3 → 1, 5 → 2, 9 → 3, 12 → 4 and 16 → 5. Write a static method that takes in an array of long values (assumed to be distinct) and returns a HashMap<Long,Integer> storing this mapping.

```
public static HashMap<Long,Integer> compress(long[] vals) {




}
```

13) (5 pts) A disjoint set of $n \geq 3$ elements, after being initialized as n separate sets has k union operations run on it where each union operation is run on a distinct pair of items. After these k operations, the disjoint set is more than one separate set. What is the maximum value of k, in terms of n? (For a simple example, when n = 3, the value of k = 1. Once we do a union between any 2 of the 3 items, the next union is forced to combine all three items into one set.)

14) (5 pts) In programming assignment #4 (Maze Magic), a regular breadth first search did not earn full credit because it would take too long on a select few cases. Explain why the fact a BFS runs in O(V+E) time on a graph with V vertices and E edges does NOT contradict the fact that for this program a BFS did not run sufficiently fast on a few cases. (Recall in the assignment that there were up to 1000 rows and 1000 columns and there were some teleportation squares.)

15) (1 pt) Today is National Blueberry Pie Day. What fruit is typically an ingredient in blueberry pie?

_____

Java Methods

Random Class

```
// Returns a pseudorandom integer in between 0 and n-1.
public int nextInt(int n)
```

Arrays Class

```
// Sorts the values in arr in non-descending order.
public static void sort(long[] arr)
```

HashMap Class

```
// Associates value with key in this map.
public void put(K key, V value)
```

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