CS 61B Fall 2023

Asymptotics, Disjoint Sets

Exam-Level 6: September 25, 2023

1 Asymptotics Introduction

Give the runtime of the following functions in Θ notation. Your answer should be as simple as possible with no unnecessary leading constants or lower order terms.

```
private void f1(int N) {
    for (int i = 1; i < N; i++) {
        for (int j = 1; j < i; j++) {
            System.out.println("shreyas 1.0");
        }
    }
}
Θ(___)
```

```
private void f2(int N) {
    for (int i = 1; i < N; i *= 2) {
        for (int j = 1; j < i; j++) {
            System.out.println("shreyas 2.0");
        }
    }
}
Θ(___)
```

2 Disjoint Sets

For each of the arrays below, write whether this could be the array representation of a weighted quick union with path compression and explain your reasoning.

	i:	0	1	2	3	4	5	6	7	8	9
A.	a[i]:	1	2	3	0	1	1	1	4	4	5
B.	a[i]:	9	0	0	0	0	0	9	9	9	-10
C.	a[i]:	1	2	3	4	5	6	7	8	9	-10
D.	a[i]:	-10	0	0	0	0	1	1	1	6	2
E.	a[i]:	-10	0	0	0	0	1	1	1	6	8
F.	a[i]:	-7	0	0	1	1	3	3	-3	7	7

3 Asymptotics of Weighted Quick Unions

Note: for all big Ω and big O bounds, give the *tightest* bound possible.

(a) Suppose we have a Weighted Quick Union (WQU) without path compression with N elements.

1. What is the runtime, in big Ω and big O, of `isConnected`?

   Ω(_____), O(_____)

2. What is the runtime, in big Ω and big O, of `connect`?

   Ω(_____), O(_____)

(b) Suppose for the following problem we add the method `addToWQU` to the WQU class. The method takes in a list of `elements` and `connects` them in a random order, stopping when all elements are connected. Assume that all the `elements` are disconnected before the method call.

```
1  void addToWQU(int[] elements) {
2          int[][] pairs = pairs(elements);
3          for (int[] pair: pairs) {
4              if (size() == elements.length) {
5                  return;
6              }
7              connect(pair[0], pair[1]);
8          }
9  }
```

The `pairs` method takes in a list of `elements` and generates all possible pairs of elements in a random order. For example, `pairs([1, 2, 3])` might return `[[1, 3], [2, 3], [1, 2]]` or `[[1, 2], [1, 3], [2, 3]]`.

The `size` method calculates the size of the largest component in the WQU.

Assume that `pairs` and `size` run in constant time.

What is the runtime of `addToWQU` in big Ω and big O?

Ω(_____), O(_____)

(c) Let us define a **matching size connection** as `connecting` two components in a WQU of equal size. For instance, suppose we have two trees, one with values 1 and 2, and another with the values 3 and 4. Calling `connect(1, 4)` is a matching size connection since both trees have 2 elements.

What is the **minimum** and **maximum** number of matching size connections that can occur after executing `addToWQU`. Assume `N`, i.e. `elements.length`, is a power of two. Your answers should be exact.

minimum: _____, maximum: _____