1 Asymptotics Introduction

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

```java
private void f1(int N) {
    for (int i = 1; i < N; i++) {
        for (int j = 1; j < i; j++) {
            System.out.println("hello tony");
        }
    }
}
Θ(____)
```

```java
private void f2(int N) {
    for (int i = 1; i < N; i *= 2) {
        for (int j = 1; j < i; j++) {
            System.out.println("hello hannah");
        }
    }
}
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.

<table>
<thead>
<tr>
<th>i:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. a[i]:</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>B. a[i]:</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>-10</td>
</tr>
<tr>
<td>C. a[i]:</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>-10</td>
</tr>
<tr>
<td>D. a[i]:</td>
<td>-10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>E. a[i]:</td>
<td>-10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>F. a[i]:</td>
<td>-7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>-3</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
Asymptotics and Disjoint Sets

3 Asymptotics of Weighted Quick Unions

For this problem, we will be addressing the asymptotics of Weighted Quick Unions! For all big $\Omega$ 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 $\Omega$ and big $O$, of isConnected?

   $\Omega(\ldots), O(\ldots)$

2. What is the runtime, in big $\Omega$ and big $O$, of connect?

   $\Omega(\ldots), O(\ldots)$

(b) Suppose for the following problem we add the method addToWQU to the WQU class. Simply put, the method takes in a list of elements and randomly connects elements together. Assume that all the elements are disconnected before the method call, and the connect method works as described in lecture.

```java
void addToWQU(int[] elements) {
    int[][] pairs = pairs(elements);
    pairs = shuffle(pairs);
    for (int[] pair : pairs) {
        connect(pair[0], pair[1]);
    }
}
```

In a bit more detail, the pairs method accepts an array and returns an ordered array of all unique pairs, where each pair is a 2 element array. For instance,

```java
pairs(new int[]{1, 2, 3})
```

would return

```java
{{1, 2}, {1, 3}, {2, 3}}
```

The shuffle method shuffles the ordering of the elements, and returns a new array. For instance,

```java
shuffle(new int[]{1, 2, 1, 3, 2, 3})
```

might return

```java
{{1, 3}, {2, 3}, {1, 2}}
```

Assume, for simplicity, that pairs and shuffle run in constant time (admittedly this couldn’t be the case, but assume so for the sake of this problem).

What is the runtime of addToWQU in big $O$? For this and all remaining subparts you may write your answer in terms of $N$, where $N$ is elements.length.

addToWQU runtime: $O(\ldots)$
For the remainder of this problem, suppose we are using the modified version of addToWQU as defined below. Note the only difference is the added if condition.

```java
void addToWQU(int[] elements) {
    int[][] pairs = pairs(elements);
    pairs = shuffle(pairs);
    for (int[] pair : pairs) {
        if (size() == elements.length) {
            return;
        }
        connect(pair[0], pair[1]);
    }
}
```

Assume the method size calculates the size of the largest connected component and runs in constant time (this can be easily implemented with adding an instance variable to the class).

(c) What is the runtime of addToWQU in big \( \Omega \) and big \( O \)?

\( \Omega(______) \), \( O(______) \)

(d) Let us define a matching size connection as connecting two trees, i.e. components in a WQU, together of matching 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 are the same size.

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: _____