Overall Asymptotic Runtime Bound for dup1

\begin{align*}
R_{\text{best}}(N) &= 2 \\
R_{\text{worst}}(N) &= \frac{N^2 + 3N + 2}{2}
\end{align*}

Give an overall asymptotic runtime bound for R as a combination of \( \Theta \), \( \mathcal{O} \), and/or \( \Omega \) notation. Take into account both the best and the worst case runtimes \( (R_{\text{best}} \text{ and } R_{\text{worst}}) \).

Then, give a few other valid runtime bounds for \( R_{\text{best}}, R_{\text{worst}}, \) and R using asymptotic notation.

Mystery

Give a tight asymptotic runtime bound for mystery as a function of N, the length of the array, in the best case, worst case, and overall.

```java
boolean mystery(int[] a, int target) {
    int N = a.length;
    for (int i = 0; i < N; i += 1)
        if (a[i] == target)
            return true;
    return false;
}
```

Q1: Give a tight asymptotic runtime bound for mystery as a function of N, the length of the array, in the best case, worst case, and overall.

Q2: Then, give a few other valid runtime bounds for \( R_{\text{best}}, R_{\text{worst}}, \) and R using asymptotic notation.
Runtime Analysis Process

Comprehending. Understanding the implementation details of a program.

Modeling. Counting the number of steps in terms of N, the size of the input.

Case Analysis. How certain conditions affect the program execution.

Asymptotic Analysis. Describing what happens for very large N, as N → ∞.

Formalizing. Summarizing the final result in precise English or math notation.

Repeat After Me...

There is no magic shortcut for these problems (except in a few well-behaved cases). Know these two summations since they’re common patterns.

\[ 1 + 2 + 3 + 4 + \cdots + Q = \frac{Q(Q + 1)}{2} \in \Theta(Q^2) \]
\[ 1 + 2 + 4 + 8 + \cdots + Q = 2Q - 1 \in \Theta(Q) \]

Strategies.
1. Find the exact count of steps.
2. Write out examples.
3. Use a geometric argument–visualizations!

Real world programs are often messy and difficult to model.

?: What’s different between these two summations?

?: How did we apply these strategies to analyze printParty?
static void f0(int N) {
    if (N < 10000)
        for (int i = 0; i < N * N; i += 1)
            System.out.println("hello");
    else
        System.out.println(N * N * N);
}

public static void f1(int N) {
    for (int i = 1; i < N; i *= i)
        System.out.println("hello");
    for (int i = 1; i < N; i *= 2)
        System.out.println("hiya");
    for (int i = 1; i < N; i += 1)
        System.out.println("hi");
}

Q: What happens when N is less than 10000? What happens when N is greater than 10000?

Q: What is the asymptotic variable in this problem?

Q1: Give the order of growth of the runtime in Θ notation as a function of N. Your answer should be simple with no unnecessary leading constants or summations.
Q1: Give the order of growth of the runtime in \( \Theta \) notation as a function of \( N \). Your answer should be simple with no unnecessary leading constants or summations.