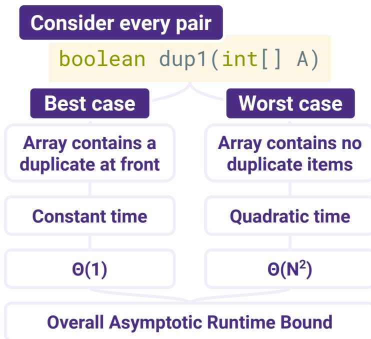
Runtime Analysis Process

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.



The reading described the implementation details for dup1 and dup2 (**Comprehension**) and introduced the idea of counting steps (**Modeling**). In this lecture, we will go in-depth on **modeling** and **formalizing**.

?: Where did case analysis come up in the reading?

Q Asymptotic Analysis vs. Case Analysis

For a very large array with billions of elements (asymptotic analysis), how is it possible for dup1 to execute only 2 less-than (<) operations?

Operation	dup1: Quadratic/Parabolic	dup2: Linear
i = 0	1	1
less-than (<)	$\boxed{2 \text{ to } (N^2 + 3N + 2) / 2}$	1 to N
increment (+= 1)	0 to $(N^2 + N) / 2$	0 to N - 1
equality (==)	1 to $(N^2 - N) / 2$	1 to N - 1
array accesses	2 to $N^2 - N$	2 to 2N - 2


```
public static boolean dup1(int[] A) {
  for (int i = 0; i < A.length; i += 1) {
    for (int j = i + 1; j < A.length; j += 1) {
      if (A[i] == A[j]) {
         return true;
      }
    }
  }
  return false;
}
```

Q1: For a very large array with billions of elements (asymptotic analysis), how is it possible for dup1 to execute only 2 less-than (<) operations?

?: What does the runtime for dup1 vs. dup2 look like if we only consider the best case asymptotic analysis? How does that result compare to the worst case asymptotic analysis?

Q Order of Growth Exercise

What is the order of growth of each function?

(Informally, what is the shape of each function for very large N?)

Function	Order of Growth
$N^3 + 3N^4$	
$(1 / N) + N^3$	
$(1 / N) + 5$	
$Ne^N + N$	
$40 \sin(N) + 4N^2$	

Big-Theta Definition

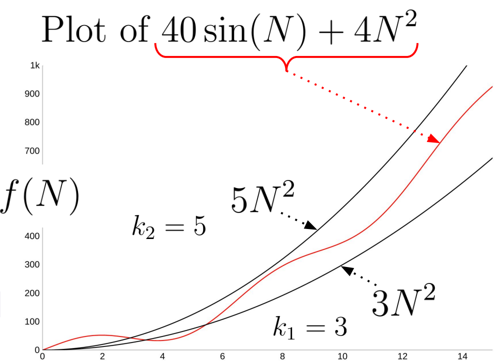
$$R(N) \in \Theta(f(N))$$

means there exist positive constants k_1 and k_2 such that

$$k_1 \cdot f(N) \leq R(N) \leq k_2 \cdot f(N)$$

for all values of N greater than some N_0.

"Very large N"



Q1: What is the order of growth of each function? (Informally, what is the shape of each function for very large N?)

?: What is a value that we can choose for N_0 according to the plot on the right?

Big-Theta Challenge

$$R(N) = \frac{4N^2 + 3N \ln N}{2}$$

Find a simple f(N) and corresponding k_1 and k_2.

$$R(N) \in \Theta(f(N))$$

means there exist positive constants k_1 and k_2 such that

$$k_1 \cdot f(N) \leq R(N) \leq k_2 \cdot f(N)$$

for all values of N greater than some N_0.

Q1: Find a simple f(N) and corresponding k_1 and k_2.

Big-O Definition

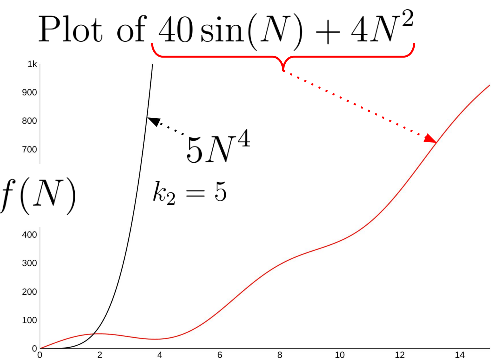
$$R(N) \in O(f(N))$$

means there exists a positive constant k_2 such that

$$R(N) \leq k_2 \cdot f(N)$$

for all values of N greater than some N_0.

"Very large N"



?: Why can we say that 40 sin(N) + $4N^2$ is in O(N^4)? Explain in terms of the formal definition of Big-O.

?: Why is it incorrect to say that 40 sin(N) + $4N^2$ is in Θ(N^4)? Explain in terms of the formal definition of Big-Theta.

Q Overall Asymptotic Runtime Bound for dup1

$$R_{\text{best}}(N) = 2$$

$$R_{\text{worst}}(N) = \frac{N^2 + 3N + 2}{2}$$

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

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

Q1: Give an overall asymptotic runtime bound for R as a combination of **Θ**, **O**, and/or **Ω** notation. Take into account both the best and the worst case runtimes (R_{best} and R_{worst}).

Q2: Then, give a few other valid runtime bounds for R_{best} , R_{worst} , and R using asymptotic notation.

Q 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.

```
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.