Goals

- Analyze the big-O of functions
- Analyze the worst-case run time of algorithms with loops

procedure *insertion sort*$(a_1, a_2, \ldots, a_n$: real numbers with $n \geq 2)$

for $j := 2$ **to** n

    $i := 1$

    while $a_j > a_i$

        $i := i + 1$

    $m := a_j$

    for $k := 0$ **to** $j - i - 1$

        $a_{j-k} := a_{j-k-1}$

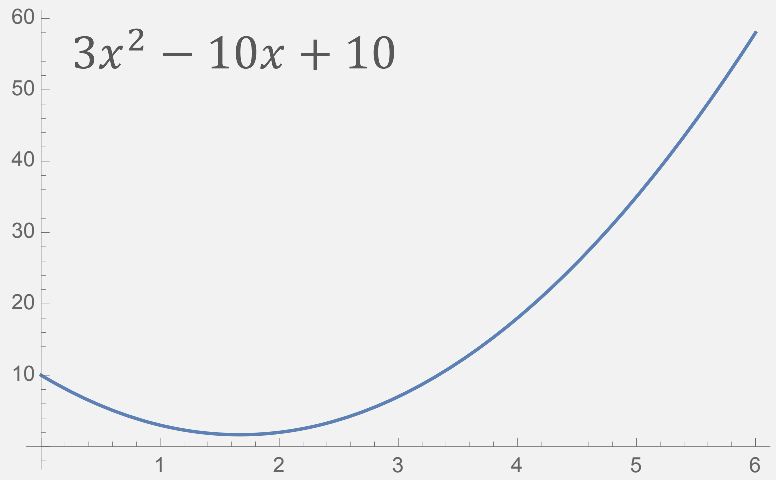
    $a_i := m$

$\{a_1, \ldots, a_n$ is in increasing order$\}$

- Do a detailed calculation of worst case # of operations
- Do a rough analysis of # of operations
- Find C, k such that

$$3x^2 - 10x + 10 = \Omega(x^2)$$

- Prove $3x^2 - 10x + 10 \neq O(1)$



Detailed Analysis

$$\#\text{ of operations} = \sum_{j=2}^{n} [\text{work done inside for loop}]$$

$$= \sum_{j=2}^{n} \left[\mathrm{A} + \sum_{i=1}^{j} B + \sum_{k=0}^{j-2} C \right]$$

$$= \sum_{j=2}^{n} [\mathrm{A} + Bj + C(j-1)]$$

$$= \sum_{j=2}^{n} [A - C + (B + C)j] = \sum_{j=2}^{n} [A - C] + (B + C) \sum_{j=2}^{n} [j - 1]$$

A, B, C represent the constant amount of operations done in loops

In the worst case, while loop runs from $i = 1$ to j

In the worst case, $i = 1$, and for loop runs from k= 0 to $j - 2$

Detailed Analysis

$$= (A - C)(n - 1) + (B + C)\left(\sum_{j=2}^{n} j - \sum_{j=2}^{n} 1\right)$$

$$= (A - C)(n - 1) + (B + C)\left(\frac{(n+2)(n-2)}{2} - n - 1\right)$$

$$= O(n^2)$$

Rough Analysis

Outer loop runs at most n times.

Two inner loops, each runs at most n times

Otherwise operations are constant.

Total is $O(n^2)$

Big-O

- Find C, k such that $3x^2 - 10x + 10 = \Omega(x^2)$

  $C = 1, k = 100$

- Prove $3x^2 - 10x + 10 \neq O(1)$

  Assume for contradiction that there exists $k, C \in \mathbb{R}^+$ such that for all $x \geq k, 3x^2 - 10x + 10 \leq C$. If we consider a value x where $x \geq 11$, $x \geq k$, and $x \geq C$, we have $3x^2 - 10x + 10 > 3x(x - 10) \geq 3C$. This contradicts that $3x^2 - 10x + 10 \leq C$ for all $x \geq k$.