\( n \) - size of input
count \# of operations
\[ f(n) \]
worst case count
\[ f(n) = 3n^2 + 2n + 2 \]
\[ g(n) = \frac{1}{2}n^3 + 16 \]

\[ \Theta(n^2) \]
\[ \Theta(n^3) \]

\[
\begin{array}{c}
\text{load } i \\
\text{load #1} \\
\text{add} \\
\text{store } i \\
i++
\end{array}
\]
Search
Linear: \( O(n) \)

Binary: \( O(\log_2 n) \)

\[ n = 0 \]
\[ n = 1 \]
\[ n = 2 \]
\[ n = 5 \]

\( \log_2(n) \)

\( O(n) - \text{linear} \)
\( O(1) - \text{constant} \)
Sort Selection

for each element (n of them)

find the min \( O(n) \)

\[
n + (n-1) + (n-2) + \ldots + 1
\]

\[
= \frac{n^2 + n}{2} = \frac{n(n+1)}{2}
\]

\[5 + 4 + 3 + 2 + 1 = 15\]

\[f(n) = \frac{n^2}{2} + \frac{n}{2} \quad \mathcal{O}(n^2)\]
Insertion Sort

$\mathcal{O}(n^2)$
Merge Sort

Split

Merge

Steps: $\log_2 n$

$O(n \log_2 n)$
Threads \( \frac{\text{a single executable unit}}{} \) {
- main
- garbage coll.
- event
}

\[
\begin{array}{cccc}
  & c_1a & c_1b & c_2 \\
 c_3 & \text{processor} & & \\
 c_4 & & & \\
\end{array}
\]
Thread Life cycle

new → start → runnable

wait → notify

wait → sleep

timeout → interrupt

I/O done → acc. lock

blocked

terminated

Runnable scheduled

ready → running

time's up
class Thread:

creating Threads

1. extend Thread
   - write a "run" method

2. implement Runnable
   - write a run method
   - pass this to a Thread object

start a thread
- call start()
- returns immediately
- causes run to be called

main:
  start
  run
do not call run