Sorting Algorithms!

Insertion Sort:

\[ \text{isort}([42, 10, 1, 6, 5]) \]

Through the wonders of recursion!

\[ [1, 5, 6, 10] \]

\[ \text{insert}(42, [1, 5, 6, 10]) \]

\[ [1, 5, 6, 10, 42] \] A new sorted list!

Write the program (both functions) on your worksheet!
Analyzing Algorithms!

def member(item, List):
    if List == []:  
        return False
    elif List[0] == item: 
        return True
    else: 
        return member(item, List[1:])

What is the **worst-case** running time as a function of the length of the input (denoted “n”)?

It’s approximately...

Asymptotic Analysis

"Asymptotic" is a fancy word!

<table>
<thead>
<tr>
<th>Function</th>
<th>Asymptotic</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3n + 42$</td>
<td>asymptotically linear</td>
</tr>
<tr>
<td>$42n + 42$</td>
<td>asymptotically quadratic</td>
</tr>
<tr>
<td>$100n^2$</td>
<td>asymptotically quadratic</td>
</tr>
<tr>
<td>$(0.1)n^2 + n + 1$</td>
<td>asymptotically quadratic</td>
</tr>
<tr>
<td>$100n^2$</td>
<td>asymptotically quadratic</td>
</tr>
<tr>
<td>$n^3 - 100n^2 + 2n + 42$</td>
<td>asymptotically cubic</td>
</tr>
<tr>
<td>$2n^3 + 10$</td>
<td>asymptotically cubic</td>
</tr>
<tr>
<td>$1/5n^3$</td>
<td>asymptotically cubic</td>
</tr>
</tbody>
</table>

Asymptotic Analysis

Simple (and not-quite-correct) rules:

1. Replace all additive and multiplicative constants by 1
2. Replace constant bases of exponents/logs by 2
3. Discard all but the highest power
   - $2^n$ beats $n^k$ for any constant $k$
   - $n^k$ beats $n^j$ for $k > j$
   - $n^i$ beats $\log n$
   - $\log n$ beats 1

Asymptotic Analysis

Those look like two awesome water-slides!
Analyzing Algorithms!

Insertion Sort:

\[
\text{isort([42, 10, 1, 65, 5])}
\]

"the magic of recursion!"

\[[1, 5, 10, 65]\]

We’re looking for the worst-case analysis!

This is amazingly cool stuff!

Let’s solve this on the board!

This Space Property of CS 5 Black

\[
1 + 2 + 3 + 4 + \ldots + (n-1) + n = ?
\]

The Alien’s Life Advice

Knowing an obscure fact isn’t proof of intelligence

...or even wisdom!
Assume—just for a moment—that the length, \( n \), is a power of 2.

Mergesort

\[ \text{msort}([42, 3, 1, 5, 27, 8, 2, 7]) \]

\[ \text{msort}([42, 3, 1, 5]) \quad \text{msort}([27, 8, 2, 7]) \]

\[ \text{merge}([1, 3, 5, 42], [2, 7, 8, 27]) \]

[1, 3, 5, 42]  [2, 7, 8, 27]

"the magic of recursion!"
Mergesort

\[
\text{merge([1, 3, 5, 42], [2, 7, 8, 27])}
\]

\[
[1, 2, 3, 5, 7]
\]
Mergesort

\[
\text{msort([42, 3, 1, 5, 27, 8, 2, 7])}
\]

\[
\text{msort([42, 3, 1, 5]) \ msort([27, 8, 2, 7])}
\]

\[
\text{merge([1, 3, 5, 42], [2, 7, 8, 27])}
\]

\[
[1, 2, 3, 5, 7, 8]
\]

Mergesort

\[
\text{msort([42, 3, 1, 5, 27, 8, 2, 7])}
\]

\[
\text{msort([42, 3, 1, 5]) \ msort([27, 8, 2, 7])}
\]

\[
\text{merge([1, 3, 5, 42], [2, 7, 8, 27])}
\]

\[
[1, 2, 3, 5, 7, 8, 27]
\]

Mergesort

\[
\text{msort([42, 3, 1, 5, 27, 8, 2, 7])}
\]

\[
\text{msort([42, 3, 1, 5]) \ msort([27, 8, 2, 7])}
\]

\[
\text{merge([1, 3, 5, 42], [2, 7, 8, 27])}
\]

\[
[1, 2, 3, 5, 7, 8, 27, 42]
\]

\text{Done!}

Let’s try it out - and let’s not even make \( n \) a power of 2!

\[
\text{msort([42, 3, 1, 6, 5, 2, 7])}
\]

\[
\text{msort([42, 3, 1]) \ msort([6, 5, 2, 7])}
\]
msort([42, 3, 1, 6, 5, 2, 7])

msort([42, 3, 1]) msort([6, 5, 2, 7])

msort([42]) msort([3, 1]) msort([6, 5]) msort([2, 7])

How “Efficient” Is Mergesort?

How big a deal is this?

Geoff’s Super-O-Matic Supercomputer:
100 billion steps/second

$n^2$ algorithm $n \log_2 n$ algorithm

$n = 10^8$ 11.5+ days