Recursion

- A process in which the result of each repetition is dependent upon the result of the next repetition.

Recursive Version

```c
int Factorial(int n) {
    if (n > 1) {
        return n * Factorial(n-1);
    } else {
        return 1;
    }
}
```

Recursion

- Every recursive algorithm can be implemented non-recursively.
  - recursion $\Leftrightarrow$ iteration
- Eventually, the routine must not call itself, allowing the code to "back out".
- Recursive routines that call themselves continuously are termed:
  - infinite recursion $\Leftrightarrow$ infinite loop
- Problem with recursive factorial implementation? Negative numbers!
- Recursion is inefficient at runtime.

Coding Recursion

- Solve the trivial "base" case(s).
- Restate general case in 'simpler' or 'smaller' terms of itself.
  - Divide and Conquer
- List Example
  - Determine the size of a single linked list.
    - Base Case: Empty List, size = 0
    - General Case: 1 + Size(Rest of List)
Types of Recursion

- Tail recursive
  - functions are characterized by the recursive call being the last statement in the function, (can easily be replaced by a loop).
- Head Recursive
  - Recursive call is the first statement (maybe after a base case check)
- Middle Decomposition

Recursion Underpinnings

- Every instance of a function execution (call) creates an Activation Record, (frame) for the function.
  - Activation records hold required execution information for functions:
    - Return value for the function
    - Pointer to activation record of calling function
    - Return memory address, (calling instruction address)
    - Parameter storage
    - Local variable storage

Function Estimation

If \( n \geq 10 \) then \( n^2 \geq 100 \)
If \( n \geq 5 \) then \( n^2 \geq 5n \)
Therefore, if \( n \geq 10 \) then:
\[
f(n) = 3n^5 + 5n + 100 < 3n^2 + n^2 + n^2 = 5n^2
\]
So \( 5n^2 \) forms an “upper bound” on \( f(n) \) if \( n \) is 10 or larger (asymptotic bound). In other words, \( f(n) \) doesn't grow any faster than \( 5n^2 \) “in the long run”.

Big-Oh Defined

- To say \( f(n) \) is \( O(g(n)) \) is to say that \( f(n) \) is “less than or equal to” \( Cg(n) \)
- More formally, Let \( f \) and \( g \) be functions from the set of integers (or the set of real numbers) to the set of real numbers. Then \( f(x) \) is said to be \( O(g(x)) \), which is read as \( f(x) \) is big-oh of \( g(x) \), if and only if there are constants \( C \) and \( n_0 \) such that
  \[
  |f(x)| \leq C |g(x)| \text{ whenever } x > n_0.
  \]
- Don’t be confused …
  - “\( f(n) \) is of Order \( g(n) \)”

The trick

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

\[
an^k + bn^{k-1} + \ldots + yn + z
\]

Some complexity classes …

<table>
<thead>
<tr>
<th>Constant</th>
<th>( O(1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logarithmic</td>
<td>( O(\log n) )</td>
</tr>
<tr>
<td>Linear</td>
<td>( O(n) )</td>
</tr>
<tr>
<td>Quadratic</td>
<td>( O(n^2) )</td>
</tr>
<tr>
<td>Cubic</td>
<td>( O(n^3) )</td>
</tr>
<tr>
<td>Polynomial</td>
<td>( O(n^p) )</td>
</tr>
<tr>
<td>Exponential</td>
<td>( O(a^n) )</td>
</tr>
</tbody>
</table>
Common Growth Curves

Does it matter?
Let \( n = 1,000 \), & 1 ms / operation.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( n ) in one day</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>1 second</td>
</tr>
<tr>
<td>( n \log_2 n )</td>
<td>10 seconds</td>
</tr>
<tr>
<td>( n^2 )</td>
<td>17 minutes</td>
</tr>
<tr>
<td>( n^3 )</td>
<td>12 days</td>
</tr>
<tr>
<td>( n^4 )</td>
<td>32 years</td>
</tr>
<tr>
<td>( n^{10} )</td>
<td>( 3.17 \times 10^{19} ) years</td>
</tr>
<tr>
<td>( 2^n )</td>
<td>( 1.07 \times 10^{301} ) years</td>
</tr>
</tbody>
</table>

Practical Curves

Best Case Analysis
- Assumes the input, data etc. are arranged in the most advantageous order for the algorithm, i.e. causes the execution of the fewest number of instructions.
  - E.g., sorting - list is already sorted; searching - desired item is located at first accessed position.

Worst Case Analysis
- Assumes the input, data etc. are arranged in the most disadvantageous order for the algorithm, i.e. causes the execution of the largest number of statements.
  - E.g., sorting - list is in opposite order; searching - desired item is located at the last accessed position or is missing.

Average Case Analysis
- Determines the average of the running times over all possible permutations of the input data.
  - E.g., searching - desired item is located at every position, for each search), or is missing.
Sorting Terms & Definitions
- Internal sorts hold all data in RAM
- External sorts use files
- Ascending Order:
  - Low to High
- Descending Order:
  - High to Low
- Stable Sort
  - Maintains the relative order of equal elements, in situ.
  - Desirable if list is almost sorted or if items with equal values are to also be ordered on a secondary field.

Comparison-Based Sorting
- Algorithms which compare element values to each other
- What is the minimum number of comparisons, required to sort N elements using a comparison-based sort?
- Is a Queue a type of sorting?

Both Sorting & Searching
- Know what each algorithm does
- How it works
- Big Oh analysis of each

N log N
- Since the depth of all leaf nodes is \( \lceil \log_2(N) \rceil \) in a comparison tree, the minimal number of comparisons to sort a specific initial ordering of N elements is \( \lceil \log_2(N) \rceil \).
- Stirling’s Approximation for \( \log_2(N!) \) can be used to determine a lower bound for \( \log_2(N!) \) which is \( O(N \log N) \).
- No comparison based sorting algorithm can sort faster than \( O(N \log N) \).

Function Template Example
```cpp
// Assumes << operator for Etype
template<typename Etype>
void PrintArray(const Etype array[], const int count)
{
    int i; //array index
    for (i = 0; i < count; i++)
        cout << array[i] << " ";
    cout << endl;
}
```

What won’t be on the test
- C-Style I/O
- Multiple function parameters
- Politics
- TV Trivia
- Sports
- Summer vacation spots
- (The stuff not covered in class)