

# For Friday

- Junit material (see website for links)
- Class is in OU 213D
- Recommended homework problems:
  - pp. 151, exs 2.9-2.11

# Program 2

- Any questions?

# Measuring Program Performance

- Given: 2 programs to accomplish a task
- Both are correct
- Which do we pick?
- Two answers
  - Smallest
  - Fastest

# Time Complexity

- Actual time computation is difficult
- Estimates are typically good enough
- Two major estimates:
  - Operation Counts
  - Step Counts

# Operation Counts

- Determine one or more operations which we believe are most important (contribute the most) to the time complexity
- We then count how many times the operation(s) occur(s).

# Step Counts

- Determine the number of steps in the program in terms of some input characteristic
- Defining a “step”
  - Any computation unit that is **independent** of the input characteristics

# Notes about Step Counts

- Need to take into account best, worst, average cases
- Take all operations into account, but inexactly.
- Purposes of complexity analysis are
  - Compare two programs that compute the same function
  - Predict growth in run time as instance characteristics change

# Comparing Program Complexities

- Suppose one program has a step count of  $100n + 10$  and one has a step count of  $3n + 3$ . Which is faster?
- Now suppose that one program has a step count of  $100n + 10$  and one has a step count of  $90n + 9$ . Which is faster?
- What if one is  $3n^2 + 3$  and the other is  $100n + 10$ ?

# Comparing Programs

- If we have two programs with complexities of  $c_1n^2 + c_2n$  and  $c_3n$ , respectively, we know that:  
 $c_1n^2 + c_2n > c_3n$  for all values of  $n$  greater than some constant
- Breakeven point

# Asymptotic Notation

- Asymptotic notation gives us a way to talk about the bounds on a program's complexity, in order to compare that program with others, without requiring that we deal in exact step counts or operation counts

# Big Oh Notation

- $f(n) = O(g(n))$  iff positive constants  $c$  and  $n_0$  exist such that  $f(n) \leq cg(n)$  for all  $n, n \geq n_0$ .
- Provides an upper bound for function  $f$ .
- Standard functions used for  $g$ :

1	constant
$\log n$	logarithmic (base irrelevant)
$n$	linear
$n \log n$	$n \log n$
$n^2$	quadratic
$n^3$	cubic
$2^n$	exponential
$n!$	factorial

# Finding Big Oh Bounds

- $3n + 2$
- $100n + 10$
- $10n^2 + 4n + 2$
- $6 * 2^n + n^2$
- 9
- 2045

# Testing

- Why test?

# Testing

- Levels
  - Unit testing
  - Integration testing
  - System testing
  - Acceptance testing
- Types
  - Black-box testing
  - White-box testing

# Planning Testing

- When do you start?
- What should test data look like?
- What is a stub?
- Test Coverage

# Drivers

# Regression Testing