

CSCE 146 Lecture 1 Software Development

- Software development phases
 - Specification of the task
 - Design of the Solution
 - Implementation fo the solution
 - Analysis of the solution
 - Testing and Debugging
 - Maintenance and evolution of the system
 - obsolescence
- algorithm
- pseudocode
- Example - Temperature Conversion Program
 - Specification
 - Design
 - Coding
 - * Specification of Java Methods
 - * Implementation of a method
 - * Implementation of a program
 -
- Javadoc
 - tool for automatically generating portions of documentation from Java code
- Analysis of algorithms
 - run-time analysis
 - How fast will the algorithm run as a function of the input size
 - what operations do you count?
- Example - Counting the steps of the Eiffel Tower
 - Specification - find number of steps in Eiffel Tower and communicate that to a friend by writing it down
 - operations to count - step down, step up, mark on paper
 - Algorithm 1 - Carry a piece of paper and place a mark for each step.
 - Algorithm 2 - No pen, no paper, but friend at top has paper and a pen.
 1. walk down and place hat on next step
 2. walk back up and report to friend to put down another mark
 3. repeat until you reach the bottom
 - Algorithm 3 - read guidebook and write down the answer in decimal

- Analysis of the Algorithms - Counting the steps of the Eiffel Tower Assuming NumSteps = 2689
 - Operations to count - step down, step up, mark on paper
 - Algorithm 1 - Carry a piece of paper and place a mark for each step.
steps down + marks on paper + steps back up = $3 * (2689)$
 - Algorithm 2 - No pen, no paper, but friend at top has paper and a pen.
 1. walk down and place hat on next step
 2. walk back up and report to friend to put down another mark
 3. repeat until you reach the bottom
 Steps down = $1 + 2 + 3 \dots + 2689$
 Carl Gauss in gradeschool
 Steps up =
 - marks on paper = 2689
 - Algorithm 3 - read guidebook and write down the answer in decimal
marks on paper - count each digit as a mark (4 marks)
- Big-O notation
 - define what an operation is
 - Formal definition of a function $f(n)$ being $O(g(n))$
 - analyze algorithm, counting operations and writing as sum of several terms
 - ignore all but dominant term
 - ignore constants
- Best-case, worst-case and average case analysis
 - best-case -
 - worst-case
 - average-case - generally requires some probabilistic assumptions
- Order of Counting Steps algorithms - assuming n steps
 - Algorithm 1: $3n = O(n)$
 - Algorithm 2: $2 * n * (n-1) / 2 + n = O()$
 - Algorithm 3: ???
- Testing and Debugging
 -
- Program Testing

-
- Properties of Good Test Data
 - you must know what the output should be
 - test inputs should include those input that are most likely going to cause errors
- Boundary values
 -
- Fully Exercising code - structural testing
 - often called structural testing
 - make sure that every line is executed at least once by your test data
 - if there is some section of code that it is possible to skip make sure there is a test set that skips this section
- Functional testing
 - often called black box testing
 - ignore internals of program
 - just test the $\text{function}(\text{input}) = \text{output}$
- Debugging Tips
 - never just start changing code in the hopes that it “might work better”
 - use debugger when possible