How to Solve Problems

// Is there a problem?
if (no_problem)
    relax;
// Is this really a problem?
else if (imaginary_problem)
    relax;
// Is this my problem?
else if (someone_else’s_problem)
    relax;
// Is this my first priority problem?
else if (not_first_priority) // Prioritize!!!
    try to solve the first priority problem first;
// Pseudocode continues on the next page.
How to Solve Problems (cont’d)

// Is this problem solvable?
// (Is it possible to satisfy the condition?)
else if (definitely_unsolvable)
    try to solve the second priority problem;
// Check the data. Garbage In — Garbage Out!!!
else if (bad_data || insufficient_data)
    make sure that you have reliable and sufficient data;

How will I recognize a solution?
// Has this problem been solved before?
if (solved_before)
    // Is the solution available to you?
    if (solution_available)
        // Can it be used?
        if (solution_can_be_used) // Check it!
How to Solve Problems (cont’d)

if (solution_works) {
    time permitting, see if you can find a more efficient solution;
    by the deadline, use the best working solution found;
}
else // i.e., if the existing known solution doesn’t work
    if (solution_can_be_fixed_quickly)
        fix the solution to make it work;
    else
        search for a working solution as described below;
else // i.e., if the existing known solution cannot be used
    if (solution_can_be_modified && used)
        use the modified solution;
else // cannot modify and use
develop an alternative solution;
else // i.e., if the existing solution is unavailable
    // Can it be reverse engineered quickly?
if (can_reverse_engineer_quickly && use)
    no comments;
    // Some problems are reverse engineering problems
else develop an alternative solution;
else // i.e., if the problem is new
    if (similar_problem_solved)
        // Make The Robustness Assumption
        see if your problem can be solved similarly;
else { // i.e., if the problem is original
    see if a simple solution presents itself; if (so) check it;
How to Solve Problems (cont’d)

// For every complex problem there is a simple solution that is wrong. George Bernard Shaw (1856-1950), Irish playwright and critic. For every complex problem, there is a solution that is simple, neat, and wrong. H. L. Mencken (1880-1956), American writer.

see if the problem can be partitioned into subproblems that can be solved separately; // Divide and Conquer!

} // end the else clause
Algorithm as a Form of Solution

- Algorithm development
  - Analyze the problem
  - Propose an algorithm
  - Test the algorithm

- Implementation
  - Code (translate into a programming language)
  - Test correctness of the algorithm and its implementation

- Maintenance
  - Use the program
  - Modify the program