RULES FOR PROGRAMMING ASSIGNMENTS

1) Programs/Homeworks are due at the **beginning of class** on the assigned due dates, put them on front desk when you come in. If you cannot make it to class have a friend turn it in for you. All homeworks are based on 40 possible points each and labs are 20 points each.

2) Homeworks turned in one class period late have three points deducted, two class periods late have six points deducted. **No homework is accepted for points after the second class period** from assigned due date. Assignments turned in one class period early have three extra points.

3) Incomplete homework will be accepted, but get as much done as possible along with a functional written algorithm and an explanation of the program's "current state" and a suggested test strategy. Remember some points are better then no points.

4) You may work together with one other person on the assignments, but DO NOT LET ONE PERSON DO ALL THE WORK. I strongly encourage you to write the program code by yourself, but you could work on the algorithm with someone else. When you turn in your homework make sure you have at the top the name of the person you worked with. If you copy someone else’s program you both will be severely penalized on points. Remember different programs will come from the same algorithm. **Each person must turn in their own separate program to be graded.**

5) Make sure all programs are commented correctly, with enough comments for clarity of operation, see examples. **DON'T COMMENT THE OBVIOUS!!**

6) Programs will be graded as follows:
   A: Comments and Program Style (25%)
      Each program should have comments which introduce the main program and each subprogram. Comments should also be used alongside lines of code that might be hard to follow. Program Style includes using good variable names and proper formatting.

   B: Testing and Program Output (25%)
      Hand in copy of program output with different cases that show the program is working correctly. In testing the logic of a program, quantity is not a substitute for quality. Program output should always contain a brief message explaining what the results are, if needed use hand written messages, no messages needed for obvious output.

   C: Program Design (25%)
      This includes things such as the design of main program and subprograms, and the proper use of procedures and functions.

   D: Algorithms (25%)
      A working program will get most of the points, but unnecessarily long programs with poor algorithms will not receive full credit.