```
┌────────────────────────────────────────────────────────────────┐
│                Project 6 Generic Grading Guidelines            │
│                            CGS 3406                             │
│                    Total possible points: 100                  │
│              Due Time: As Announced in Project Write-Up        │
│         NO EXTENSIONS AND NO LATE PROGRAMS WILL BE ACCEPTED     │
└────────────────────────────────────────────────────────────────┘
```

*Note: these are generic grading guidelines, intended to give you an idea of
how the project will be graded and what kinds of things the grader will be
looking at. These guidelines are tentative and are subject to change. You
will be provided with specific guidelines when you get your graded project
back. Before you turn in your program, be sure to read the style guidelines
handout on the course web site and utilize all information you have been given
in lecture, recitation, on the web site and via email.*

Working program/Correct Results: worth 50% of total points

   syntax errors - does not compile (up to -50)
   syntax warnings
   compiles but will not run (link errors etc.)
   unreachable code (e.g. if/then always false)
   errors in expressions, calculations and equations
   missing evaluations for special cases
   passing parameters using call by value when they should be passed
     using call by reference
   not reading input data correctly
   run with wrong data set(s) or file, if required
   did not echoprint input, if required
   missing output which was required
   runtime error (program crashes)

Documentation/Style : worth 25% of total points

Use the Programming Style Guidelines for C++ handout as the guide for
requirements of style, formatting, documentation, etc.

   main program header comment not sufficient
   header comments - each function
     must include a (not too) brief description of what the
     function does, its input/output, parameters, etc.
   lack of comments explaining code
   lack of comments describing variables, constants, etc.
     ALL variables/constants declared *EVERYWHERE* must be commented
   lack of comments on { } pairs (each, where needed)
   inaccurate or meaningless comments
   bad program format aesthetics (indentation, capitalization,
     insufficient white space in code around sections,
     operators etc.)
   bad output format aesthetics (e.g. not enough white space,
     meaningless output messages, unlabeled output)
   bad output variable formats (e.g. unformatted numbers)
   poor formatting of output in tables
   meaningless or overly-abbreviated identifiers
   did not declare appropriate named constants (using const)

Efficiency / program structure: worth 25% of total points

   unnecessary code (each) (e.g. repeating nearly identical code)
   unnecessary conditions on if
   unnecessary { } s (each)

exception: certain situations where the structure of
      a control structure is clarified by using an extra { }
    using repeated if/then (instead of if/then/else or switch)
    code inside an if (or switch) that belongs outside
    violating one-entrance one-exit rule when not justified,
      example, while(1) loops
    use of break, continue, or return when not justified
    use of goto
    code inside a loop that belongs outside the loop

    passing parameters using call by reference when they
      should be passed using call by value
    passing parameters by reference when they
      should be by const reference
    using any global variables
    using only one function, main
    poor breakdown of tasks among main and other functions
      (general lack of good structure and clear flow and/or
       lack of functional cohesion - each function should perform
       one task or several very closely related tasks)

    not using the specific data structure(s) required by the write-up, if any
    not setting up classes and class members as specified in write-up
    not using the specific algorithms(s) required by the write-up, if any

Miscellaneous Notes:

  - when output is incorrect, points are taken off for the cause
    of the errors (e.g. errors in expressions, equations, missing
    equations, etc.)
  - don't deduct more points than an area is worth (e.g. don't
    deduct more than 35% of possible points for documentation
    etc.)
  - points will be taken off for the use of C constructs which
     are not compatible with the focus of this course, which is for
     students to learn the conventional use of the standard
     C++ language (not to learn C). For example, points will be
     deducted for using scanf or printf instead of cin and cout, for
     using pointer parameters instead of reference parameters, and for
     using #define instead of the const qualifier to declare named
     constants.
     Students must use only standard ANSI C++ header files which are
     discussed in the course textbook or in class as being part of
     ANSI standard C++. Standard C++ header files do include C++
     headers derived from C such as cmath, but they do not include
     old C headers such as math.h. These headers are also not
     standard and cannot be used: stdafx.h and conio.h.

Automatic zeroes:

  - not turned in by due deadline
  - the following situations, if an attempt at "cheating" appears
    to have occurred
      two or more programs which are substantially identical
      if output is turned in, it could not have come from the
        program turned in

Late penalties:

This assignment must be turned in by the deadline due time or it
will receive a ZERO grade. That is, no late programs will be
accepted, and no extensions will be given. THIS IS <u>NOT</u> THE
SAME POLICY AS FOR PREVIOUS COURSE PROJECTS !!!

Missing/Other Item Penalties:

- not submitted electronically and correctly to Blackboard site
- required file missing (this results in a zero score)
- more than one file submitted
- required file not named correctly as specified in write-up
- note: file turned in must compile and run correctly using
  Microsoft Visual C++ 6.0 Express 2008 compiler

CORRECT RESULTS: see sample solution

REGARDING FSU COMPUTER COMPETENCY REQUIREMENT

This assignment is assessed according to the FSU Computer Competency Requirement.
As such students must demonstrate (1) competent use of a discipline-specific
software package: Microsoft Visual Studio C++ and (2) the ability to perform simple
transactions using the web/Internet: obtaining all of the assignment materials from
the class web site and downloading the required file(s), and turning in the
completed work correctly to the class web site.