

Project 4 Generic Grading Guidelines
COP 3014

Total possible points: 100

Due Time: As Announced in Project Write-Up

Turn in "ON TIME": by 11:59 pm on due date

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 class web site and utilize all information you have been given in lectures, recitations, the course web site and via email.

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

- syntax errors - does not compile
 - using required compiler and platform (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(s), if required
- did not echoprint input, if required
- missing output which was required
- runtime error (program crash)

Documentation/Style : worth 25% of total points

* NOTE: starting with this project, the percentage for documentation is 25%, efficiency/structure also 25%

Use the Programming Style Guidelines for C++ course handout as a 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 (e.g. in-line comments)
- lack of comments describing variables, constants, etc.
 - ALL variables/constants declared everywhere must be commented
- lack of comments on { } pairs (where needed)
- inaccurate or meaningless comments
- bad program format esthetics (indentation, capitalization)
- insufficient white space in code
 - (spacing around lines, sections, operators etc.)
- bad output format esthetics (e.g. no 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 and use appropriate named constants

Efficiency / program structure: worth 25% of total points

unnecessary code (each) (e.g. repeating nearly identical code)
unnecessary conditions on if which cannot be justified
unnecessary { } s (each)
 exception: certain situations where the structure of
 a control structure may be 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
use of return, break, continue when not justifiable
use of goto (any)
use of any code or structures which violate one-entrance-one-exit
 principle e.g. "while(1)" loops
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 when they should have been declared as
 local variables
using only one function, main
poor breakdown of tasks among main and other routines
 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
all functions must have correct prototypes

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 current ANSI standard C++. Standard C++ header files do include C++ headers derived from C such as cmath, but they do not include older 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 late 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:

Take off 20% if turned in by 11:59 pm 48 hours following the original due date

Missing Item Penalties:

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

CORRECT RESULTS: see sample solution
