CS-139 Policy on Collaboration

One of the problems faced by students everywhere is understanding the degree of collaboration that is permitted in the completion of school assignments. This document is intended to help computer science students understand some of the issues, particularly as they apply to computer programs.

Collaboration in General

When discussing collaboration during the work on an assignment, a number of issues arise: task, source, scope, degree, and citation.

Tasks.

School assignments typically consist of a number of tasks. For an assignment to write an English paper, different tasks might include:

- understanding what the assignment is asking to be covered in the paper,
- adopting a basic theme
- developing the paper's outline
- writing the actual sentences
- selecting fonts and other word processing features to design the look of the manuscript
- checking for spelling and grammar errors

For each task, the degree of permitted collaboration may differ (a student may NOT be permitted to obtain any outside help for writing the words but may be able to use a spell checker to check for spelling errors).

Source.

Assistance for an assignment can be obtained from a number of sources:

- class textbooks
- other books
- students in the class
- other people outside the class
- the class instructor
- another faculty member
- the class teaching assistant
- someone at the JMU computing help desk
- an internet chat room
- other internet sources
- a computer software package.

Assistance from some sources (the course instructor) may be permitted while assistance from other sources (other students) may not be.

Scope.

The scope of collaboration deals with the extent to which the completed work is a product of collaboration. Did the collaboration just concern a single sentence in a 20 page effort? Or was the design of 25% of a computer program affected.

Degree.

While scope deals with the quantity of assistance received, degree deals with the quality or depth of assistance received. While in some cases degree may be difficult to define, there are two easily defined levels of degree: **None** and **Copy**. **None** means no assistance in any form was received. **Copy** means to work was done by someone else and copied. Some other levels that might be defined are:

- **General:** The assistance does not lead to the inclusion of any specific material in the submitted effort. For something to be considered general assistance it must have the following characteristics:

(1) the usefulness of the assistance would not be significantly diminished if it was delivered independent of the effort to complete the assignment and could not be acted on until 24 hours later (without any notes), and
(2) if the same assistance was given to multiple students, the ultimate product of that assistance would be different for each student.

It is never general assistance if to provide it, one had to look at a student's work. Thus, if for a paper on love in Shakespeare comedies, one could probably consider the suggestion to "contrast romantic love with love between family members" as general assistance. However, help with the wording of a particular paragraph or suggestions to reference particular passages of Shakespeare's work would not be general assistance.

- **Paraphrase:** Someone else's idea is re-written to say or do the same thing. For computer programs, paraphrased code is code that uses a similar looping and branching structure and a similar set of variables. Paraphrased design results in the software system using similar modules with similar interactions between modules.
- **Isomorphic-copy:** Someone else's work is copied but certain proper names are changed or synonyms are used. Also the order of sentences or paragraphs might be altered. An isomorphic copy of the above might read as follows. "Another's effort is copied but the order of paragraphs or sentences might be changed. Also synonyms are used or certain proper names are modified." In isomorphic code, variable names, comments, indentation and spacing, and the order of statements and declarations may be different, but the underlying meaning of the code is identical.

Citation and Plagiarism

Some forms of assistance may be permitted **ONLY** if they are properly cited. Typically, papers require paraphrased work of others to be referenced. Copied passages must be quoted. Any work produced as a team effort should have all the team members listed as co-authors in ALL copies of the work submitted. Failure to properly cite sources is considered plagiarism.

It is important to understand the difference between plagiarism and permitted collaboration. Plagiarism can be defined to be "the deliberate copying, writing or presenting as one's own the information, ideas or phrasing of another person without proper acknowledgment of the true source" (http://www.jmu.edu/judicial/handbook.html, 08/10/02). If one includes the work of another without proper citation, one is guilty of plagiarism and may be subject to honor system sanctions. If one includes the work of another with proper citation but the use of another's work is not permitted in completing the assignment (one submits an English paper that was copied in its entirety from someone else but the original author is properly cited), there is no plagiarism but the work submitted does not satisfy the assignment requirements. Although honor system penalties will not apply, the grade for the assignment may be reduced, even to the extent that no credit is given, because the requirements were not met.

The offense of plagiarism is far more serious than the offense of failing to meet an assignment's requirements. Whenever using the information, ideas, or phrasing of another person in one's work, cite the original author unless you have received explicit instructions that citing the original author is unnecessary. Including a citation where it is unnecessary does no harm. Failing to include a citation where it is necessary can get you expelled from school.

Typical Computer Science Project Tasks

To understand how collaboration affects computer science programming projects, one must understand what the different tasks are. In relation to these tasks, each computer science professor may establish his/her own rules relating to permissible collaboration either on a per project basis or for the course as a whole.

The tasks typically encountered when working on programming assignments are as follows.

Assignment Specification

The first thing a student must do in working on an assignment is understanding what the assignment is asking them to do. One must understand the assignment specifications before one can satisfy them.

Program Specification

In many courses, this step is the same as the previous step. Program specification answers the question, what is it the software you are creating must do? That information may have been specified in the assignment specification or it may be something you will have to determine. With detailed program specifications, one should be able to determine what the program's output should be for any input.

Program Design

Large problems are often broken down into a collection of interacting problems that can be solved independently. The design of a solution involves decomposing the problem into these smaller units and defining the interaction between units. It is in many ways similar to developing an outline for an English paper (although much more detailed). In beginning programming courses, program design may be performed by the instructor.

Code Creation.

Next, one must create the program code. This is typically involves a process of refinement where one begins by creating a code "outline" (similar to an outline for a paper), possibly using some form of psuedo-code. The statements in the outline (or psuedo-code) are progressively made more precise until they evolve into actual statements in the programming language.

Compile and Link

Code that has been created must be entered into the computer. The programmer must know how to use basic system commands or menu options for manipulating files, how to use an editor to enter code into a file, how to compile and link the source code into an executable, and possibly how to electronically submit the program to the course instructor. Compilation typically results in syntax errors which must be corrected. This step is analogous to using a word processor to enter the text of and English paper and to check and correct spelling and grammar mistakes.

Testing and Debugging

Once there is an executable, it must be tested. With testing comes the discovery of bugs which must be removed. With testing and correction of bugs, one must step back to earlier tasks in the process. The resolution of a bug encountered during testing could lead back to changing a program design for one or more of the solution components.

It should be clear this is not a linear process where one progresses directly through all the tasks. Insights revealed in a task frequently require one to backtrack to any of the previous tasks.