Policy on Cheating and Collaboration[1]

Students are encouraged to read the university policy regarding cheating. If you have any questions or concerns, then please contact the course instructor to discuss the matter. *When in doubt, ask the instructor.*

Certain forms of collaboration are beneficial for helping everyone understand the material better and overcome small stumbling blocks. But of course blatant copying of projects and homeworks does not benefit anyone, and undermines trust.

In this course, certain forms of collaboration are acceptable and beneficial to your learning process. Direct engagement with the material, in discussion or in doing programming projects is the best path to understanding. Therefore, activities such as discussing the projects to understand them better, helping locate conceptual bugs, and discussing lecture and textbook content are acceptable. In general, collaboration not only helps you get the job done, it teaches you how to explain inchoate ideas to others. This is why we permit discussion of the problems between students.

But it is also important that your collaborations be balanced and fair with respect to other students. If you misuse the opportunity for collaboration, you will do poorly in the course. But *what you hand in must be your own work.*

The challenge is how to distinguish between acceptable discussion (and software engineering is almost always a group activity!) and unacceptable sharing or appropriation of work done by others. You might find it helpful to keep this analogy in mind. Imagine that you are taking an English course on short stories and that you have been given the assignment to write a short story. It is acceptable collaboration to discuss your story ideas with others. You may try out your ideas by telling your stories to others, discussing plot development, character development, and so on. But when it comes down to writing your story, *must write your own words and your own story*. You might be able to read Neal Stephenson and even understand thoroughly why his stories are great; but this doesn't mean that you may copy his words.

It is, of course, difficult at times to know if what you are doing will be considered cheating. If you are unsure whether an action you are contemplating would be considered cheating, then contact the instructor. Again, *when in doubt, ask the instructor.*

Here are some examples of **acceptable collaboration**:

- Clarifying ambiguities or vague points in class handouts, textbooks, or lectures.
- Discussing or explaining the general class material.
- Providing assistance with Java or Eclipse APIs, in using the system facilities, or with editing and debugging tools.
- Discussing the code that we give out on the assignment.

[1] Adapted from Professors Jonathan Aldrich and William Scherlis from Carnegie Mellon University.

- Discussing the assignments to better understand them.
- Getting help from anyone concerning programming issues which are clearly more general than the specific project (e.g., what does a particular error message mean?).
- In general, verbal collaboration is OK.

Now for the dark side. As a general rule, if you do not understand what you are handing in, or if you have written the same code as someone else, you are probably cheating. If you have given somebody some code, so that it can be used in that person's project, even if cleverly rewritten, you are probably cheating. In order to help you draw the line, here are some examples of clear cases of **cheating**:

- Copying (program or assignment) files from another person or source, including retyping their files, changing program identifiers (package names, class names, field names, method names, variable names, etc.), copying code without explicit citation from previously published works (except the textbook), etc.
- *Allowing* someone else to copy your code or written assignment, either in draft or final form.
- Getting help that you do not fully understand, and from someone whom you do not acknowledge on your solution.
- Writing, using, or submitting a program that attempts to alter or erase grading information or otherwise compromise security.
- Copying someone else's files containing draft solutions, even if the file permissions are incorrectly set to allow it.
- Lying to course staff.
- Reading the current solution (handed out) if you will be handing in the current assignment late.

Technical Steps for Prevention

To help prevent cheating, you *must* read-protect your own directory in which you develop your projects. You must also *lock* your workstation, and *password-protect* your screensaver if you step away from a machine in a lab where other students enrolled in the class may be working.

Final Points

We believe that very few students actually plan to cheat. Usually it arises as a desperate act due to poor planning, overload, or other circumstances.

If you are feeling desperate enough to consider cheating, please talk to the course instructor about alternative strategies for addressing the challenges that are bringing you to this point. There are many ways to resolve issues without taking on personal and ethical risks with potential lifetime consequences!