

# Computer Science and the Stanford Honor Code

*Dr. Patrick Young, CS193C*

*This document is based on material written by Professor Eric Roberts.*

Since 1921, academic conduct for students at Stanford has been governed by the Honor Code, which reads as follows:

## THE STANFORD UNIVERSITY HONOR CODE

- A. The Honor Code is an undertaking of the students, individually and collectively:
  - (1) that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
  - (2) that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.
- B. The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid as far as practicable, academic procedures that create temptations to violate the Honor Code.
- C. While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

In the Computer Science Department, we take the Honor Code seriously and expect you to do the same. The good news is that the vast majority of you will do so. The bad news is that all historical evidence indicates that some students in computer science will submit work that is not their own, shortchanging not only their own learning but undermining the atmosphere of trust and individual achievement that characterizes Stanford's academic community. Each year, the Computer Science Department accounts for somewhere between 20 and 40 percent of all Honor Code cases, even though our courses represent only about seven percent of the student enrollment.

The purpose of this handout is to make our expectations as clear as possible in the hope that we will reduce the number of Honor Code violations that occur. The basic principle under which we operate is that each of you is expected to submit your own work in this course. In particular, attempting to take credit for someone else's work by turning it in as your own constitutes plagiarism, which is a serious violation of basic academic standards.

From the attention that the department pays to the Honor Code, some of you will get the idea that any discussion of assignments is somehow a violation of academic principle. Such a conclusion, however, is completely wrong. In computer science courses, it is usually appropriate to ask others—the TA, the instructor, or other students—for hints and debugging

help or to talk generally about problem-solving strategies and program structure. In fact, I strongly encourage you to seek such assistance when you need it. The important point, however, is embodied in the following rule:

**Rule 1: You must indicate on your submission any assistance you received.**

If you make use of such assistance without giving proper credit, you may be guilty of plagiarism.

In addition to providing proper citation—usually as part of the comments at the beginning of the program—it is also important to make sure that the assistance you receive consists of general advice that does not cross the boundary into having someone else write the actual code. It is fine to discuss ideas and strategies, but you should be careful to write your programs on your own. This provision is expressed in the following rule:

**Rule 2: You must not share actual program code with other students.**

In particular, you should not ask anyone to give you a copy of their code or, conversely, give your code to another student who asks you for it. Similarly, you should not discuss your algorithmic strategies to such an extent that you and your collaborators end up turning in exactly the same code. Discuss ideas together, but do the coding on your own.

The prohibition against looking at the actual code for a program has an important specific application in computer science courses. Developing a good programming assignment often takes years. When a new assignment is created, it invariably has problems that require a certain amount of polishing. To make sure that the assignments are as good as they can be, Stanford's department—like most others in the country—reuses assignments over the years, incorporating a few changes each time to make them more effective. The following rule applies in all computer science courses:

**Rule 3: You must not look at solution sets or program code from other years.**

Beyond being a clear violation of academic integrity, making use of old solution sets is a dangerous practice. Most assignments change in a variety of ways from year to year as we seek to make them better. Each year, however, some student turns in a solution to an assignment from some prior year, even though that assignment has since changed so that the old solution no longer makes sense. Submitting a program that solves last year's assignment perfectly while failing to solve the current one is particularly damaging evidence of an Honor Code violation.

Whenever you seek help on an assignment, your goal should be improving your level of understanding and not simply getting your program to work. Suppose, for example, that someone responds to your request for help by showing you a couple of lines of code that do the job. Don't fall into the trap of thinking about that code as if it were a magical incantation—something you simply include in your program and don't have to understand. By doing so, you will be in no position to solve similar problems on exams. The need to understand the assistance you receive can be expressed in the following rule:

**Rule 4: You must be prepared to explain any program code you submit.**

In accordance with Stanford's judicial policy, we are required to tell you that we may use plagiarism detection tools to help identify possible violations. We archive all submissions, both from this quarter and past quarters, and cross-compare for unusual resemblance. We do not target specific students, all assignments are subject to the same scrutiny. Any similarity detected by the tools is then examined more closely by our staff and, where appropriate, cases are referred to Judicial Affairs. The tools are very adept at identifying all variants of improper

collaboration, from major to minor. Last year, these tools uncovered about two dozen cases of plagiarism in the CS106 classes, I hope that we have far fewer this year.

**Rule 5: All submissions are subject to automated plagiarism detection.**