International Journal of Computer Science Education in Schools, Oct 2017, Vol. 1, No. 4
ISSN 2513-8359

How to Reduce Cheating in an Introductory Computer Programming Course?

Jacob Sukhodolsky

Saint Louis University, Department of Computer Science, USA
jacob.sukhodolsky@slu.edu

DOI: 10.21585/ijcses.v1i4.13

Abstract

The problem of Computer Science students' cheating in their homework assignments so far has been handled mainly through administrative punishment of the cheaters. The success of such an approach depends to a large degree on the ability of the instructor to recognize the fact of cheating, which is a complicated task. With a large number of students taking the course, identifying the cheaters sometimes requires considerable time. The author of this paper suggests a method of solving the cheating problem by changing the course grading policy. The suggested approach emphasizes the importance of regular checking of students' understanding the course material.

Keywords: cheating, syllabus, homework, assignment, quiz.

1. Introduction

Over the years, I have been teaching an introductory Computer Programming course for non-computer science majors more than fifteen times. Often, I wondered how some students who received no less than 'B'-s for their homework assignments could score below 'C-' in their mid-term and final exams. After taking a closer look, I realized that they were cheating by copying their classmates' homework assignments. Cheating on computer science assignments has been a major concern for a large number of universities, such as U. of Florida, Texas San Antonio, Princeton, Stanford, Carnegie Mellon, and MIT, for over twenty years (Alcantara, 2012; Bidgood, 2017; Krieger, 2010; Phillips, 2014; Wagner, 2000). Many instructors believe that cheating is so popular among computer science students because of the high frustration levels students experience trying to get a program to run (Krieger, 2010). A computer program is considered running if it returns an adequate result for any conceivable input. Developing a running program makes writing and testing it a tedious task, which may require a considerable amount of students' time. Copying and editing someone else's program is a shortcut, saving students time they can devote to their other courses.

The strategy many universities employ in their fight against cheating in computer science assignments consists of two steps:

1. Detection of cheating
2. Dealing with cheating incidents

The simplest way to check if a student has cheated is to compare his/her program to those of other students. Finding numerous similarities between programs is a sign that cheating has taken place. The described approach has serious weaknesses. For instance, if a programming assignment is relatively simple, and all the students of the class use the textbook example of a program similar to the one assigned, there is a strong possibility that a number of programs written independently will look similar to each other. Another problem is a large size of the class, which requires the

instructor to spend a prohibitive amount of time comparing students' code. Some educators recommend using software, like Moss (Krieger, 2010; Phillips, 2014), which detects program similarities. Good results have been obtained with Caveon, software that exposes cheating by using statistical anomalies (Gabriel, 2010). Another statistical method used to determine if cheating has taken place is described in (Silverman, 2015).

If a student is identified as a cheater, the instructor has to choose the appropriate course of action. Every course syllabus contains a clause stating that cheating of any kind is not tolerated (See the Appendix). Some universities even expel cheaters. A number of educators, including myself, disagree with such a policy. William Murray writes: "I tend to see 'cheating' as a symptom that something is wrong in the system, in pedagogy" (Murray, 2010, p. 2). In my courses, I have never implemented the cheating policy in its strictest form, even when I caught a student cheating. I would talk to the students (the one who copied and the one whose assignment was copied) and let them know that the next time they violate the rules they will deal with the disciplinary committee. Typically, that has worked. I personally believe that it is not worthwhile to waste valuable professor's time trying to catch students cheating and then administering a punishment. In one of the eldest published papers on the topic, Shaw, Jones, Knueven, McDermott, Miller, & Notkin (1980) are also of the opinion that preventing cheating is much more preferable than trying to detect cheating and take action against the students involved. The Rutgers University School of Arts and Sciences includes a section on how to prevent cheating into their guidelines for new instructors. In it, they recommend using grading schemes that are designed to limit opportunities for cheating.

This paper presents the introductory computer science course cheating prevention strategy first introduced in (Sukhodolsky, 2015). It demonstrates how the described strategy reduces the problem of cheating in homework assignments to minimum. The paper first analyzes the problem of cheating. It examines the process of teaching and the conventional methods of evaluating students' understanding of the material. It demonstrates then how cheating can be reduced by modifying the course syllabus. The paper also contains the results of applying the proposed method. At the end, the author presents the results of a survey he used to get feedback from the students who took his course in Spring 2017.

2. Analysis of the Problem

2.1 What is Cheating and why it is Bad

There have been published a considerable number of articles analyzing the problem of cheating. Chuda, Navrat, Kovacova, & Humay found that "100% of staff and 94% of students consider copying, modifying, or presenting another's source code as plagiarism" (Chuda et al., 2012, p. 24).

The most popular cheating cases include the following (Mary Shaw, et al., 1980, p.75):

- Submitting someone else's slightly edited code as your own
- Allowing a fellow student to submit your homework as his/her own (in some cases, they even forget to change the programmer's name)
- Working as a team on an assignment that is supposed to be done individually
- Copying a part of someone else's exam
- Stealing an exam or solution from the instructor.

According to (Mary Shaw et al., 1980), the majority of instructors notice that most of cheating takes place with homework assignments rather than exams. Therefore, the last two cases of cheating are left out of the scope of this paper.

Some papers focus on the harm caused by cheating. The worst consequence of cheating is that by the end of the course a cheater neither understands the course material, nor possesses the programming skills he/she has been taught. Cheating is harmful because if even a few students cheat without being detected, the others feel demotivated (Chuda, et al., 2012). Cheating is also bad because the students who copy their homework without even trying to understand the problem and the program used to solve it tend to put blame on the instructor in their course evaluations.

2.2 Applying the System Approach

Let us look at the problem from the System Control theory point of view. An instructor is trying to manage a class of students in such a way that they learn and understand the course material. At the start of the course, the students are provided with the course syllabus. Throughout the semester, the students sit through lectures and are assigned homework, tests, and quizzes. The instructor gets feedback regarding how successfully his/her goals are being

accomplished from the homework assignments, tests, or quizzes returned by the students. The instructor grades the students' homework, tests, and quizzes (Fig. 1).

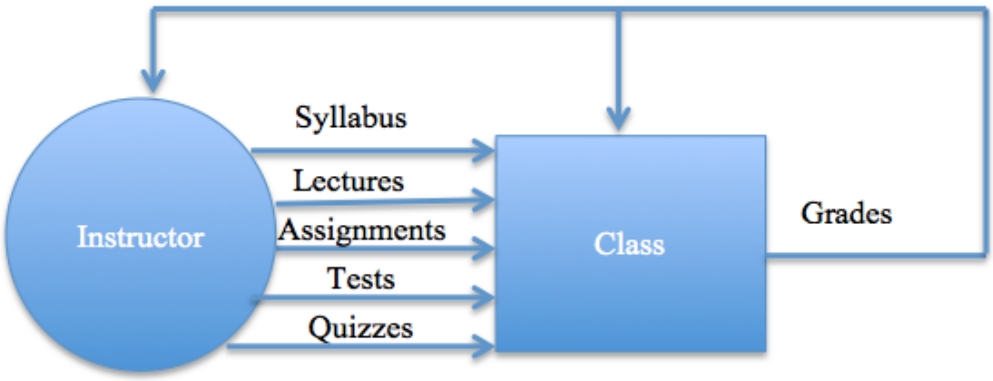

Figure 1. Schematic presentation of the teaching process

The instructor evaluates each student's performance based on his/her grades. It does not look logical if a student does great with the homework but poorly in the tests or quizzes, especially if the class assignments are open book. Although some students perform poorly under time pressure, generally the students getting low grades on the quizzes are the ones who copy homework assignments without trying to understand them. As a result, the students fail to demonstrate in their class assignments the knowledge and skills they were supposed to acquire in their homework assignments.

The instructor should help such students by explaining to them the material they have failed to understand. It works in many cases, but only if the students are willing to come to the instructor with questions. Unfortunately, very few students use that option. The more effective way for the instructor is to look critically at the course syllabus, the way the course material is presented, and the recommended textbook to determine possible changes/improvements in the way the course is taught. This paper concentrates on the course syllabus, particularly on the part of it that describes the course grading policy.

2.3 Analysis of the Course Syllabus

To be more specific, let us use as an example the syllabus of the computer programming course taught to freshmen who were not majoring in computer science (see Appendix). The course consists of two parts. In the first one, the instructor teaches how to use MATLAB to write simple scripts that include input/output statements, branching statements, loops, and functions. Students also learn how to write programs including file write and read operations. In the second part of the course, students learn some basic concepts of C++. For many years, the course grading policy was:

Homework (35%)
Quizzes (15%)
Midterm exam (25%)
Final exam (25%)

The homework assignments here were weighted so high because of the instructor's belief that the students would better learn computer programming if they were successful with the homework assignments over all the topics of the course. Usually, the assignments were fairly simple. Each of them emphasized the use of a particular programming concept. Students knew that if they submitted all or at least the majority of the homework assignments, a course-passing grade would be guaranteed. Therefore, they tried to submit their assignments, even incorrect ones, on time (late submissions were not accepted). Such an approach led to copying assignments from other students. A student whose goal was just to get a passing grade on the course did it easily. For instance, a student who earned 56 points out of 100 (which is an 'F') for his final exam could get over 75 points (a 'C+') overall. Such a low final exam grade

clearly demonstrated the student's poor knowledge and understanding of the course material and was not consistent with a 'C+' final course grade.

3. Proposed Problem Solution

The straightforward method for solving the problem of cheating is to try to catch the cheaters, give them the 'F' course grade, and get the University Student Behavior Committee involved. The result is that the guilty are punished but the ultimate goal of the course – having students learn computer-programming skills - is not achieved. Another approach is to "organize the course and computer usage to avoid situations in which students might be pressured or tempted to cheat" (Shaw et al., 1980, p. 73) At the same time, the educational quality of the course should not be compromised.

The approach proposed here requires changing the course syllabus. First of all, the syllabus must clearly define what constitutes cheating. For example,

- Copying from a fellow student.
- Copying from the Web.
- Working together with a group of students on a non-group assignment.

Each student is allowed to get assistance from their class mates when working on non-group assignments. The assumption was that such a practice will help improving students' understanding of the material. Each student is also required to report the names of the students he/she worked with on the assignment. All the students who are part of a group working on the assignment receive the same grade.

After one or more assignments on a given topic, a quiz on the topic is given to the class (each student is quizzed individually). The quiz reveals the names of the students who have actually understood their homework assignments. A similar method is described in (Fraser, 2014). Homework assignments are necessary because - by doing them - students develop new skills. The quizzes are important because they help the instructor evaluate students' basic understanding of the course topics. Quizzes therefore are more important and should have a larger weight than assignments. The above weights should be changed to, for instance, the following:

Homework (20%)
Quizzes (35%)
Midterm exam (20%)
Final exam (25%)

Such a course grading policy is in line with the Rutgers University cheating prevention guidelines that recommend instructors to limit homework assignments to no more than 10% of the total course grade, at the exception of special courses where regular practice is deemed essential and which cannot be replaced by in-class quizzes or exams. It is clear that with such a grading scheme, excellent homework assignment grades cannot compensate for poor quiz grades. Each point of the quiz grade average is 1.75 times more valuable than a point of the homework assignment grade average. In other words, earning a homework grade average of 100 is equivalent to only around 57 points of quiz grade average. It means that the overall course grade will be poor if the quiz grades are low – assuming similarly low midterm and final exam grades. The probability of a student to receive high midterm and final exam grades while performing poorly on quizzes is very low, since students who fail elementary quizzes will hardly succeed at the more challenging midterm and final exams. After each semester, the instructor should critically inspect the grades to determine if they reflect the students' knowledge and skills acquired in the course. If necessary, the grading scheme should be adjusted.

4. Results

The described approach was implemented in the Fall 2015, Spring 2016, and Spring 2017 semesters. In the Fall 2015, only one section of the course was taught and the method was not fully implemented, as the students were not quizzed on all the course topics. But in the Spring of 2016, when two sections of the course were offered, the method was used fully on both. As expected, the correlation between the quiz grades and the final course grades was pretty high – 0.91 for the first section and 0.78 for the second. Also, the calculated average size of the homework assignment teams was 2.46 and 2.64, correspondingly. Interestingly, the first section's average course grade was 77.07, while the average course grade of the second section (with the larger average team size) was 81.48.

Judging by the course evaluations, the method was well received by the students. Valuable information was also obtained by analyzing students' answers to a survey at the end of the Spring 2017 semester. Each of the 41 students who participated in the survey was asked the following questions.

1. Did you do your Scientific Programming homework assignments individually or with other students? (Individually/With others)

   18 of 41 students sometimes did their homework with other students, sometimes individually.
   12 students did their homework individually.
   11 students did their homework with other students.

2. Did you choose to do it because:
   a) It helped you improve your understanding of the material
   b) It resulted in a smaller number of mistakes
   c) It took less time
   d) Other (please explain)

   The students who did their homework sometimes with others and sometimes individually did it because
   - It helped them improve understanding of the material (16 students)
   - It resulted in a smaller number of mistakes (12 students)
   - It took less time (5 students)

   The students who did their homework individually did it because
   - It helped them improve understanding of the material (8 students)
   - It resulted in a smaller number of mistakes (1 student)
   - It took less time (5 students)
   - Didn't know anybody in the class (2 students)

   Those who did it with others did it because
   - It helped them improve understanding of the material (10 students)
   - It resulted in a smaller number of mistakes (4 students)
   - It took less time (2 students)

   One student, who mostly worked on homework assignments individually, wrote: "When I worked with others, it was because I didn't understand the topic, and working with others helped me see what I wasn't understanding."

   As a total, 34 out of 41 students believed that working on their homework with other students helped improving their understanding of the material.

3. Do you believe it is fair to assign quizzes the largest weight in the course grading policy? (Yes/No)

   38 students answered 'Yes', while 3 students answered 'No'. A few students supplied specific comments:
   "I would suggest having quizzes carry less weight and having them more frequently."
   "Yes, there are plenty of quizzes and they are relatively short. It's nice to not have a grade almost completely depend on a couple exams."
   "Yes, it makes us study the material constantly."

4. On the scale 1 to 10 evaluate the complexity of the course quizzes (1 corresponds to the easiest)?

   The answers varied from 2 to 10. 5.74 was the average.

5. Were the quizzes related to the homework assignments? (Yes/No)

   All the students answered 'Yes'. A few more specific answers:
   "The quizzes were always fair and correlated to the homework and topics that were covered in class."
   "But they were easier than the homework. The homework was often difficult."

6. Were the quizzes helpful in your preparation for the Midterm and Final exams? (Yes/No)

   37 students answered 'Yes'.
   4 students answered 'No'.

One student answered: "Yes, because they served as good indicator of what I need to focus on while studying."

5. Conclusions and Future Research

This paper presented an approach that can be used to reduce cheating in introductory computer science courses. Instead of using the traditional methods of "fighting" cheating through administrative actions against the students found guilty of cheating, the proposed approach suggests changing the course syllabus in a way that does not forbid students to work on their homework assignments together with others. In other words, doing homework together with other students is no longer classified as cheating. As a result, the probability of cheating is reduced to the minimum. The majority of the students are very unlikely to cheat during a quiz, with the instructor looking at them. As it is demonstrated in the article, homework assignments alone cannot reliably evaluate the skill level and understanding of a student. Instead, periodic quizzes are an effective tool of verifying students' level of understanding of the course material. The author used the proposed approach in the Fall 2015, Spring 2016, Fall 2016, and Spring 2017 semesters. The survey conducted among the students who took the course in Spring 2017 provides an indication that the proposed approach helps students to better understand every homework assignment and its solution.

The method works because it encourages collaborative learning, which was found to be beneficial to many students (Chamillard, Kim, & Braun, 2000).

The approach may also benefit the instructor by reducing the time spent on assignment grading. If student X reported he/she collaborated with students Y and Z, there is no need to run the programs submitted by Y and Z. On the other hand, the number of quizzes should be equal, at least, the number of topics covered in the course. The amount of time mentioned above may not be realized if, for instance, X reported he/she collaborated with students Y and Z but Z reported collaborating with only Y. In such a case, the instructor will have to clarify who actually collaborated with whom. Unfortunately, not all the students let the instructor know that they worked on the assignment with someone else. In such cases, the instructor's time is wasted on grading identical assignments.

Some instructors do not believe that weighing tests more than homework assignments is a good solution because "assignments test different things than exams" (Phillips, 2014, p. 5). Their reason is that there are students who do much better on the assignments than they do on the written exams. Yet, many of them are good computer programmers. The proposed grading strategy does not contradict that opinion. The weights assigned to the midterm and final exams are the same as that of the homework assignments. The quizzes are much more valuable component of the course grade than homework assignments because they check students' understanding of the basics fundamental for program writing.

References

Alcantara, C. (2012). University of Florida Students Caught Cheating on Computer Science Projects. *The Independent Florida Alligator*. March, 13, 2012.

Bidgood, J., Jeremy B. Merrill. (2017). As Computer Coding Classes Swell, So Does Cheating. *The New York Times*. May, 29, 2017.

Chamillard, A.T., Kim A. Braun. (2000). Evaluating programming ability in an introductory Computer Science course. *SIGCSE '00 Proceedings of the thirty-first SIGCSE technical symposium on Computer science education*, 212-216.

Chuda, D., Navrat, P., Kovacova, B., Humay, P. (2012). The issue of (software) plagiarism: a student view. *IEEE Transactions on Education, 55(1), 22-28.*

Gabriel, T. (2010). Cheaters find an adversary in technology. *The New York Times, Dec. 27, 2010.*

Fraser, R. (2014). Collaboration, collusion and plagiarism in Computer Science. *Informatics* In Education, 13(2), 179-195.

Krieger, L.M. (2010). Stanford finds cheating — especially among computer science students — on the rise. *San Jose Mercury News, June 8, 2010*.

Murray, W.H. (2010). Cheating in Computer Science. *Ubiquity, 2*.

Phillips, P., Cohen, L. (2014). Convictions of plagiarism in computer science courses on the rise. *The Daily Princetonian, March 4, 2014.*

Rutgers University School of Arts and Sciences Advice for New Instructors on How to Prevent Academic Dishonesty. [Online]. Available: http://www.finmath.rutgers.edu/academics-finmath/for-new-instructors/149-

advice-on-how-to-prevent-academic-dishonesty
Shaw, M., Jones, A., Knueven, P., McDermott, J., Miller, P., Notkin, D. (1980). Cheating policy in a Computer Science department. *ACM SIGCSE Bull, 12(2), 72-76.*
Silverman, M.P. (2015). Cheating or coincidence? Statistical method employing the principle of maximum entropy for judging whether a student has committed plagiarism. *Open Journal of Statistics, 5, 143-157.*
Sukhodolsky, J. (2015). Cheating Prevention in Computer Science Courses. *2015 Proceedings of the Information Systems Education Orlando, Florida USA, 371-378.*
Wagner, N. (2000). Plagiarism by Student Programmers, http://www.cs.utsa.edu/~wagner/pubs/plagiarism0.html

Appendix: Syllabus

CSCI-145-01 Introduction to Scientific Programming
Syllabus

Instructor: Dr. Jacob Sukhodolsky
Email: sukhodpj@slu.edu
Office location: Ritter Hall 226
Office hours: MWF 10:15 – 12:30 am, or by appointment
Phone: 314-977-2443
Course title: Introduction to Scientific Programming
Course number: CSCI-145-01
Course begins: Wednesday, January 12, 2015
Course ends: Wednesday, May 12, 2015
Location: MDH 1066L
Meeting time(s): MWF 9 am - 9:50 am

Course description:
In this class you will learn elementary computer programming concepts, with an emphasis on problem solving and applications to scientific and engineering fields. Topics include data acquisition and analysis, simulation and scientific visualization. Examples with be taken from real-world engineering and scientific applications. Students will acquire skills in

- using basic programming constructs: loops, conditions, functions, input and output
- data representation: basic data types, strings, arrays, multi-dimensional arrays
- incorporating the existing software libraries and packages into their programs
- simulating real-world events.

MATLAB
You may purchase a student version of MATLAB if you wish (~$100). Matlab is available in the labs in MDH and online off our departmental server called turing.slu.edu.

C++
We will run C++ compiler off the departmental server.

Required Textbook
"MATLAB Programming with Applications for Engineers" by Stephen J. Chapman

Optional Textbook
Any C++ textbook published after 2000.

Reading
Read the textbook for the required reading before lectures, and study them more carefully after class. Please note that all the required readings are fair materials for exams. These materials may not be fully covered in lectures -- lectures are intended to motivate as well as provide a road map for your reading, given the limited lecture time we may not be able to cover everything in the readings.

Evaluation
Course grades will be assigned according to the following weighting formula:

Homework (35%)
Quizzes (15%)
Midterm exam (25%)
Final exam(25%)

Midterm Exam: Friday, March 6, 2015

Final Exam: Wednesday, May 6, 2014, 8:00 – 9:50 a.m.

Late Policy
Homework is due till midnight following the day it is due. No late homework will be accepted. Graded work will be returned to you in a timely manner.

Student Integrity & Academic Honesty
Cheating of any kind will not be tolerated. Any assignment or exam that is handed in must be your own work. However, talking with one another to understand the material better is strongly encouraged. Recognizing the distinction between cheating and cooperation is very important. If you copy someone else's solution, you are cheating. If you let someone else copy your solution, you are cheating. I won't distinguish between the person who copied a solution and the person whose solution was copied. Both people will be treated as cheaters.
If someone dictates a solution to you, you are cheating. Everything you hand in must be in your own words, based on your own understanding of the solution.
If someone helps you understand the problem during a high-level discussion, you are not cheating. We strongly encourage students to help one another understand the material presented in class, in the book, and general issues relevant to the assignments. When taking an exam, you must work independently. Any collaboration during an exam will be considered cheating.
Any student who is caught cheating will be given an F course grade and referred to the University Student Behavior Committee. Many students think they can get away with cheating and will not be caught. But it is much easier to spot cheater than you might think. Please don't take that chance -- if you're having trouble understanding the material, please let us know and we will be happy to help. Students are expected to have read and to abide by the University statement on Academic Integrity available on page 56 of the Saint Louis University's undergraduate catalog.

Student Success Center Academic Support Statement:
In recognition that people learn in a variety of ways and that learning is influenced by multiple factors (e.g., prior experience, study skills, learning disability), resources to support student success are available on campus. Students who think they might benefit from these resources can find out more about:
Course-level support (e.g., faculty member, departmental resources, etc.) by asking your course instructor.
University-level support (e.g., tutoring/writing services, Disability Services) by visiting the Student Success Center (BSC 331) or by going to www.slu.edu/success.
Students who believe that, due to a disability, they could benefit from academic accommodations are encouraged to contact Disability Services by phone (314-977-8885) or via the Student Success Center Webpage (http://slu.edu/x24491.xml). Confidentiality will be observed in all inquiries. Course instructors support student accommodation requests when an approved letter from Disability Services has been received and when students discuss these accommodations with the instructor after receipt of the approved letter.