Teaching Academic Honesty in CS50

David J. Malan
Harvard University
malan@harvard.edu

Brian Yu
Harvard University
brian@cs.harvard.edu

Doug Lloyd
Harvard University
lloyd@cs50.harvard.edu

ABSTRACT

We aspire to teach academic honesty in CS50 at Harvard University not only by addressing academic dishonesty when it occurs but by addressing it before it does. By way of communication, course- and campus-wide awareness of policy, just-in-time prompts, interventional conversations, and problem sets have we tried to preempt submission of plagiarized work. But few interventions have had significant or lasting effects on the number of instances thereof. Most impactful has been the addition of one sentence to the course's syllabus, a "regret clause" that encourages students to come forward within 72 hours of some dishonest act on their part, before the course itself is even aware. While we might zero the work in question in such cases, we commit to not escalating the matter further to the university's honor council, where the outcome might instead be admonishment, probation, or even required withdrawal from the university itself. We instead advise students on how best to move forward and connect them as needed with support structures on campus for academics and mental health. Since 2014 have 89 students invoked the clause, between 1% and 3% of the course's student body each year.

CCS CONCEPTS

• **Information systems → Near-duplicate and plagiarism detection**; • **Social and professional topics → Codes of ethics**; *Computer science education*;

KEYWORDS

academic dishonesty, academic honesty, code, ethics, honor council, plagiarism, policy

ACM Reference Format:
David J. Malan, Brian Yu, and Doug Lloyd. 2020. Teaching Academic Honesty in CS50. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366940

1 INTRODUCTION

A perennial topic of discussion at Harvard University, as at SIGCSE, is academic dishonesty, instances of plagiarism whereby students submit work that is not, in some way, their own. Indeed, as recently as SIGCSE 2018 was a birds-of-a-feather flock on the topic aptly subtitled, "the Discussion Continues." [8] Our own course, CS50, has the unfortunate distinction of referring to the university's honor council, most every year, more cases for disciplinary action than any other at the university. It has been frequently assumed, and sometimes even stated, that we must therefore be doing something wrong. But we claim, on the contrary, that we are doing something right. In fact, to our knowledge, we are among the few courses at the university that systematically look for evidence of academic dishonesty. And, as computer scientists, we are perhaps especially equipped with software to detect it.

We have resisted suggestions that we simply change the course's problem sets (i.e., programming assignments) each year. Quite often do our cases involve code shared only among current students anyway. We have instead preferred to improve our problem sets each year, as by clarifying specifications, incorporating past answers to frequently asked questions, fixing bugs, and adding tests. We daresay the net result is that they get better with time. And we have also preferred not to rely more on, say, proctored exams for students' grades, as students spend most of their time on (and learn most from, we hope) the course's problem sets.

Even so, we have focused in recent years not so much on punitive outcomes but on educational ones instead. Most impactful has been our introduction of a "regret clause" to the course's syllabus, which asks students to come forward, on their own, within 72 hours of submitting work that is not, in some way, their own. In return, we meet students halfway, scheduling a conversation with them to discuss first what happened and then how to move forward. Any penalty thereafter is limited to zeroing the work in question; we commit to not escalating the matter further to the university's honor council, where the outcome might instead be admonishment, probation, or even required withdrawal from the university itself. We then consider the matter behind us.

We have aspired, of course, to reduce the frequency of plagiarism itself, as by discussing the topic at term's start with students and disclosing to students how we detect such. We have offered late days and even late-night, no-questions-asked extensions. And we have even introduced a problem set on document similarity for which students implement an approximation of the software we ourselves use for detection. But no intervention to date has yielded a significant, or lasting, reduction. Instead, we seem to have identified via our regret clause a demographic previously unidentified and, thus, unsupported: students who, to their credit, after submitting work not their own, wished to take responsibility therefor. Remarkably, among the dozens of students who have come forward under that clause since 2014 to admit some act of dishonesty, few of those acts were even detected by software. Had those students not come forward on their own, most would not have appeared on our radar at all. Yet invocations of that clause have led to heart-to-heart talks, referrals for mental health, and, ultimately, teachable moments for an otherwise not-previously-reached demographic. But that same

SIGCSE '20, March 11–14, 2020, Portland, OR, USA

ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366940

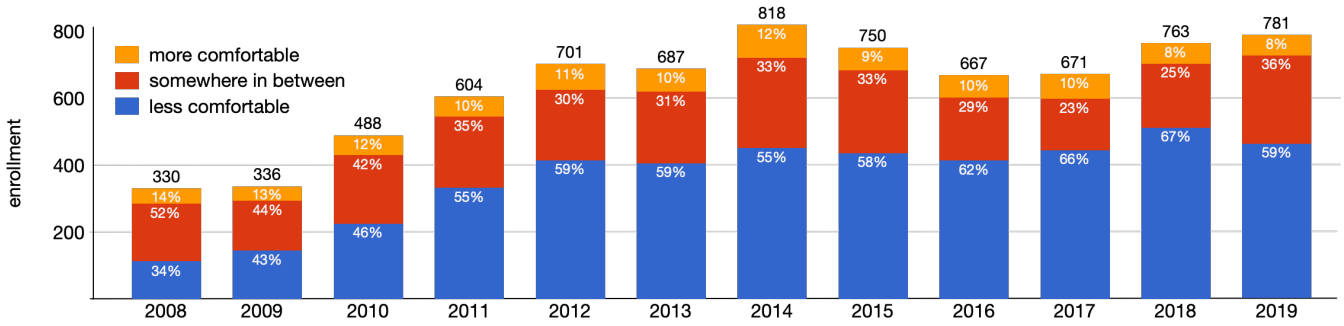

Figure 1: Enrollment in CS50 since 2008, categorized by comfort level. At term's start, students are asked whether they consider themselves among those less comfortable, more comfortable, or somewhere in between with respect to computing. Those less comfortable are now the course's largest demographic.

6,091,800 pairwise comparisons
↓
1,200 matches
↓
116 matches after human review
↓
38 students referred to honor council

Figure 2: The process by which students' submissions are reviewed for evidence of academic dishonesty resembles a funnel. In Fall 2019, 6,091,800 pairwise comparisons yielded 1,200 matches, which were then whittled down to 116 matches; after further review, 38 students were referred to the university's honor council.

clause has also contributed to an uptick in the number of cases referred to the university's honor council for disciplinary action, in part because we now feel more comfortable referring cases after students have had an opportunity to take ownership themselves but have chosen not to do so.

In the sections that follow, we present our experience and results. We begin first in Section 2 with background on the course, including its process and policy. We then detail in Section 3 our interventions, present and past. We explore related work in Section 4. And in Section 5, we conclude and recommend.

2 BACKGROUND

CS50 is Harvard University's introduction to computer science for majors and non-majors, an amalgam of courses generally known as CS1 and CS2 elsewhere, taught primarily in C, followed by Python. The course is the university's largest, with nearly 800 students enrolled in Fall 2019. As elsewhere [16], enrollment has trended upward in recent years, with students (self-described as) less comfortable with computing the course's largest demographic, per Figure 1. The course meets weekly for lectures, in which concepts are introduced, followed by sections, in which concepts are reviewed in smaller groups with 20 or fewer students. Sections are led by the course's teaching fellows (TFs), who also hold office hours, optional opportunities for students each week for one-on-one help. The course's workload is non-trivial, with most students spending more than 12 hours per week on problem sets.

2.1 Policy

The course's syllabus acknowledges that interactions among classmates and others can facilitate mastery of the course's material but cautions students that there remains a line between enlisting the help of another and submitting the work of another. The syllabus then characterizes both sides of that line, elaborating in detail on acts considered "reasonable" and "not reasonable" while providing examples of each. (See Appendix for policy in full.) But the essence of the course's policy is this:

> Generally speaking, when asking for help, you may show your code to others, but you may not view theirs.

To be sure, two students could contrive a scenario in which they both ask each other for help, but the full policy prescribes guardrails that disallow.

2.2 Process

Each week, upon submission of a problem set, the course cross-compares every submission against every other as well as against past years' submissions and known repositories online. To automate that process, the course has used Moss [1] as well as ETector [12] but more recently has transitioned to its own open-source (and extensible) alternative, `compare50`, per the Appendix. Those weekly comparisons tend to yield dozens of pairwise matches, ranked roughly from more similar to less similar, from which a senior member of the course's staff culls a subset of matches that their human eyes, from experience, judge suspicious as well. A more senior member of the staff then refines that list further in consultation with the course's instructor. The course's instructor then decides, in consultation with senior staff, which students to refer to the university's honor council for formal review of the evidence. The course is informed later of the outcome, which might be none, local sanctions, admonishment, probation, or required withdrawal [2].

The process, then, is a funnel of sorts, per Figure 2. In Fall 2019, for instance, approximately six million pairwise comparisons in software yielded 1,200 matches, all of which were reviewed by

human eyes, 116 of which (10%) were flagged for further review, after which 38 students were ultimately referred to the university's honor council.

On average since 2008, the course has referred nearly 4% of its student body to the university's honor council, albeit with high variance, and the annual percentage has trended upward (even if 2016 is removed as an outlier), per Figure 3. Not only does that trend coincide with rising enrollment, particularly among less-comfortable students, the deltas also seem to coincide with of some of our own interventions.

3 INTERVENTIONS

While punitive outcomes in cases of academic dishonesty might have some educational value in the form of lessons learned, we have long aspired to preempt the cases themselves. Early on did we offer "late days," one- or few-day extensions that students could grant themselves, in hopes of relieving some mid-semester pressure that might otherwise build up. But we have also, in recent years, endeavored to be even more proactive than reactive by way of these interventions.

3.1 Communication

Among our earliest attempts to reduce the frequency of academic dishonesty was, quite simply, communication. We began to discuss the topic head-on in lecture at term's start, around the time that students would submit their first problem sets. Not only would we discuss the course's policy, we would also share the number of students referred to the honor council in the previous year in hopes that the number alone would deter.

We also began to include the course's policy not only in the course's syllabus but in each of the course's problem sets as well. While we did not expect students to read the policy weekly, we hoped that its reappearance might at least serve to remind.

Communication about policy and numbers alone, though, did not seem to impact students' behavior. While it's true the course did not refer any students to the honor council in 2009, per Figure 3, we suspect that was a result that one year of spending less time on the process. Indeed, the percentage of students referred returned to 2% in 2010 and even rose to 5% in 2011.

In 2012, though, that percentage fell to just 1%, per Figure 3, even though enrollment itself rose, per Figure 1. It's worth noting that, in 2012, we not only discussed the course's policy in class, we also, for the first time, provided specific (anomymized but representative) examples of past behavior that had gotten students in trouble. For instance, we disclosed to students that software can still detect plagiarized code, even when one student changes

```
int ytimes = 0;
int xtimes = 0;
```

to

```
int timesy = 0;
int timesx = 0;
```

as had recently happened. It's possible we thus taught students how to evade detection that year. But the software we used detects more than transpositions alone; it detects structural similarities as well, so quite a few matches reached human eyes still. And just as we suspect we spent less time in 2009 reviewing submissions, it's possible we spent more time in 2011. Indeed, 2012's percentage (1%) was back in line with 2008's (2%), 2010's (2%), and 2013's (2%).

3.2 Raising Awareness

But we suspect awareness on campus might also explain the drop from 2011 (5%) to 2012 (1%), insofar as more students in 2012 might have known classmates who had been disciplined in 2011. In fact, we saw an even more marked drop from 2016 (10%) to 2017 (4%). Not only might that particular delta be the result of coverage of 2016's numbers by the university's newspaper, the course also invited the university's dean of undergraduate education to speak to the matter in class at term's start in 2017, which might have raised awareness more. Moving forward, we might begin to ask students via surveys at term's start just how familiar they are with CS50's history of cases so that we might measure awareness year over year. Of course, the honor council's proceedings are confidential, so those involved might not even disclose their involvement to others.

3.3 Prompts

It's worth noting that we also integrated a prompt into the course's command-line submission software in 2017, a just-in-time effort to remind students to consider their action:

> Keeping in mind the course's policy on academic honesty, are you sure you want to submit these files?

Students must explicitly type `yes` in order to submit their work. It's possible that prompt might have helped reduce 2016's 10% to 2017's 4%, but even before that prompt's introduction was the percentage at 4% (in 2014).

3.4 Regret Clause

After referring so many students to Harvard's honor council over so many years, however, we noticed a pattern. All too often were students' acts the result of late-night panic, a combination of little sleep, much stress, and one or more deadlines looming. We hypothesized that at least some of those students might very well regret those acts the next morning after some sleep and reflection. But, at the time, there was no well-defined process via which they could take ownership of the situation, short of waiting to see if their act would be noticed by term's end. To be fair, there was also no process preventing students from coming forward. But with the potential penalty so high (e.g., required withdrawal from the university), it's not surprising that few, if any, ever did on their own.

And so we added one sentence to the course's syllabus in 2014:

> If you commit some act that is not reasonable but bring it to the attention of the course's heads within 72 hours, the course may impose local sanctions that may include an unsatisfactory or failing grade for work submitted, but the course will not refer the matter for further disciplinary action except in cases of repeated acts.

While we considered other numbers of hours, we felt that three days were adequate opportunity for students to rest and reflect, even during their busiest times.

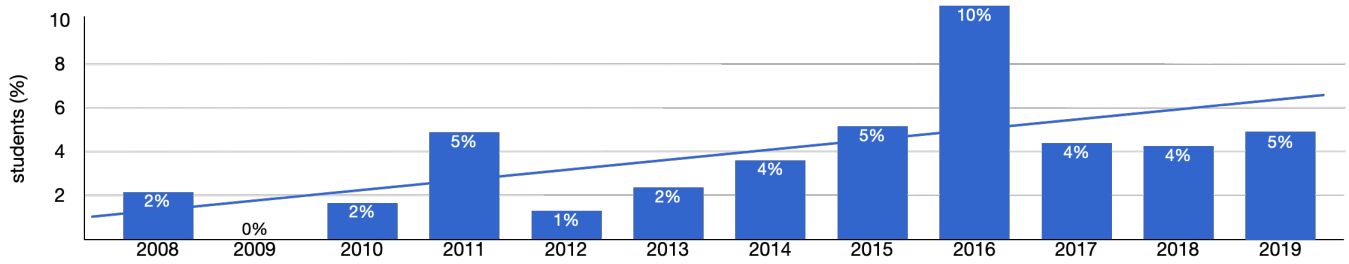

Figure 3: The percentage of students (as a function of enrollment) that the course refers to the university's honor council each year has been variable but upward-trending since 2008.

The clause was not without controversy. Officially, all matters of academic dishonesty must be referred to the university's honor council, and so its members were among the first with concerns. Among the council's goals, understandably, is to ensure that all such matters are handled consistently across campus as well as to detect dishonesty in multiple courses by the same student. We noted, however, that we would still refer evidence that we ourselves detected as usual. We were only proposing to handle internally matters that students themselves brought forward, for which, historically, to our knowledge, there was no precedent on record. And we emphasized the educational intent of the clause. Ultimately, the council was willing to let us proceed but asked that we add the clause's final words, "except in cases of repeated acts," so that the council might at least intervene in such cases. The university's newspaper also weighed in, characterizing the clause as a get-out-of-jail-free card. Of course, by its own definition, it wasn't.

The clause debuted in Fall 2014. Not only was it communicated to students by way of the syllabus, we also discussed it in lecture at term's start.

By term's end, 19 students out of 818 (2%) had invoked the regret clause at some point during the term, per Figure 4. In each case did the instructor meet with the student to discuss what had happened. In most of those cases, the work in question was zeroed, but no further action was taken. Indeed, we took care to communicate that we considered the matter behind us. In six of those cases, we took no action at all, as we did not feel that the student had crossed any lines; they were unnecessarily worried.

In many cases, both in 2014 and since, those discussions have become heart-to-heart, if tearful, talks, not only about the act of dishonesty itself but about extenuating circumstances in the student's life that somehow contributed. In such cases have we then connected the student with additional resources on campus for academic support and mental health.

In 2015, 26 students out of 750 (3%) invoked the regret clause. Far fewer then did in 2016, 7 out of 667 (1%), the result, we suspect, of less messaging from us, which we then corrected in 2017, at which point invocations again rose to 18 out of 671 (3%). Since then have invocations again fallen, to 11 out of 763 (1%) in 2018 and 8 out of 781 (1%) in 2019. In both years, though, did the course remind students, each week, of the regret clause's availability, prompting them on a web-based form to acknowledge that "I am aware that I can invoke [the course's] regret clause specifically." We suspect

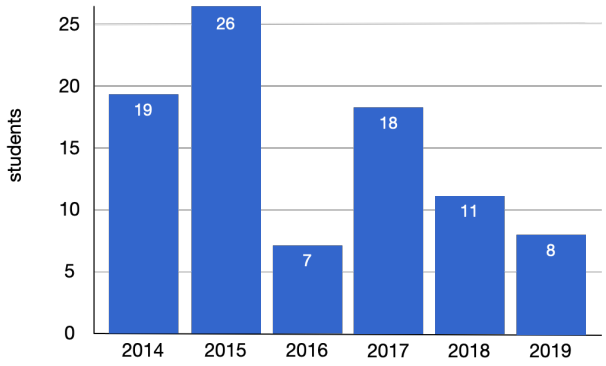

Figure 4: Numbers of students since 2014 who have invoked the course's regret clause.

those weekly reminders might have reduced the actual frequency of acts that students might otherwise later regret.

While we thought that the regret clause might reduce our number of cases by allowing students to come forward before we ourselves detected, our number of cases actually increased, from 2% in 2013 to 4% in 2014, 5% in 2015, and 10% in 2016. While factors like insufficient communication and awareness those years might partly explain, we also found ourselves more comfortable in 2014 onward referring cases because we felt we had, via the regret clause, already met students halfway. The honor council asked, too, that we not wait for patterns of behavior to emerge among students but that we refer any evidence of dishonesty to them weekly after each problem set so that they might intervene educationally sooner. In years past, we tended to wait until term's end to review students' submissions.

We also introduced into our process in 2015 that funnel, per Section 2.2, albeit toward term's end, whereby multiple members of the staff began to review students' submissions. By 2017, that funnel instead happened weekly. In years past, only the course's instructor would review, and, indeed, usually at term's end. While the funnel ensured that cases would only proceed if three pairs of eyes had suspicions instead of just one, it also increased the total number of hours spent on reviews and, in turn, the thoroughness thereof.

That combination of regret clause and weekly funnel, we suspect, explained 2016's 10%.

3.5 Interventional Conversations

In 2017, though, we transformed our own process from binary to ternary. Rather than decide between referring and not referring a case based on some evidence, we began to have interventional conversations with those students whom we did not feel had crossed a line but who, based on their submissions, had potentially come close, as by discussing their own solution in too much detail, even without showing. Our goal was not to investigate for more evidence but, rather, to understand the students' own workflow and help them navigate the syllabus's guardrails. We suspect those conversations explain, in part, the reduction in cases from 2016 (10%) to 2017 (4%).

3.6 Brink Clause

Not all interventions have been successful, however. In 2018, we added a "brink clause" to the course's syllabus, inspired by similar language by Moretti [10] at Princeton, which we adapted as follows:

> If push comes to shove, and you reach a breaking point this term, particularly late at night, and fear you're about to commit some act that is not reasonable, you may, up to the last moment before you cross that point of no return, email the course's heads to say that you see no other way out but invoke this clause. (And then go to sleep!) We will then meet with you and, together, get you back on track. For your honesty and your return from the brink, we will waive any penalty for lateness.

This particular clause was invoked by students 38 times that term, but most invocations (79%), we felt, were but extension requests guised (ironically) as honesty-related. Indeed, based on correspondence with students, we concluded that 11%, at most, were truly consistent with the intent of the clause. Two invocations led to tutoring, but the nature of others proved ambiguous. Even so, we honored all invocations without question. But we removed the clause in 2019.

3.7 Problem Sets

We also introduced in 2017 a problem set on document similarity itself for which students write code that compares text files for characters, words, and sentences in common, thereafter visualizing the results. The problem set itself is not framed as being about plagiarism, but it's inspired by the software we ourselves use. Not only did we craft the problem set for its algorithmic value, we also hoped to clarify for students just how easy it is to compare code. To be fair, the problem set happens late in the term, by nature of its prerequisites, and has thus not had a measurable effect on students' behavior. As recently as 2018, we referred six students to the university's honor council based on evidence that their code for that very problem set was not, in fact, their own.

4 RELATED WORK

Discussion of academic dishonesty in literature abounds but focuses more on courses' reduction and detection thereof and less on students' response thereto. Indeed, we have found little discussion, save anecdotes, of student-initiated interventions akin to our own regret clause and analysis thereof.

Fraser [6], however, does provide an overview of academic dishonesty in computer science and information technology, proposing how to establish boundaries among students while still supporting collaboration. Riedesel et al. [14] offer a guide to existing policies and how to apply and revise them, while Simon et al. [15] similarly offer guidelines. Harding et al. [7] examine five separate institutions and identify predictors of cheating.

As recently as SIGCSE 2019, meanwhile, did Mason et al. [9] discuss how they reduced instances of plagiarism in Georgia Tech's online MS program by clarifying the university's policy and assessing students on the same, interventions similar to some of our own. At SIGCSE 2017, by contrast, Pierce and Zilles [13] considered partnering students to be one intervention but found that it did not reduce the incidence of plagiarism. Dick et al. [5], though, have emphasized cultural changes, proposing that students are less likely to plagiarize if they themselves understand the educational goals of their work. Albeit in the context of psychology students, Curtis et al. [4] have advocated for online modules that teach academic integrity, much like Owens and White [11] have advocated for in-class exercises and online assessments on the same.

And, albeit in the context of first-year business students, Cronan et al. [3] find that "Higher moral obligation (greater sense of guilt) will correspond with a lower/weaker intention to violate academic integrity." Even though students who invoked our regret clause had already technically violated academic integrity, a higher moral obligation might nonetheless explain why they quickly came forward.

5 CONCLUSION

In CS50 at Harvard, we aspire to teach academic honesty, not only via reactive, punitive lessons but by proactive, educational interventions, among them communication, awareness, prompts, conversations, and problem sets. Among our goals, too, are fairness to other students whose submitted work is entirely their own and, ultimately, a more honorable citizenry. Admittedly, that latter goal's outcome is harder for one course to measure.

Most impactful locally has been the course's regret clause. Though not without controversy early on, it remains the course's most meaningful, ongoing intervention. We recommend its adoption by others.

While the course continues to refer an average of 4% of its students per year to the university's honor council, we hope that the educational impact of even punitive lessons is long-lasting. We concede that reduction to 0% might be naive, as there might always be students to teach. Albeit proof by example, just recently did the course's instructor receive a note from one alumnus, apologizing for long-ago behavior, appreciative of the lesson, even though not at the time.

APPENDIX

Open-Source Software

The open-source software that we now use to cross-compare submissions of code, `compare50`, can be found at https://github.com/cs50/compare50. It includes an extensible API.

Full Policy

The course's philosophy on academic honesty is best stated as "be reasonable." The course recognizes that interactions with classmates and others can facilitate mastery of the course's material. However, there remains a line between enlisting the help of another and submitting the work of another. This policy characterizes both sides of that line.

The essence of all work that you submit to this course must be your own. Collaboration on problem sets is not permitted except to the extent that you may ask classmates and others for help so long as that help does not reduce to another doing your work for you. Generally speaking, when asking for help, you may show your code to others, but you may not view theirs, so long as you and they respect this policy's other constraints. Collaboration on the course's quizzes and test is not permitted at all. Collaboration on the course's final project is permitted to the extent prescribed by its specification.

Regret clause. If you commit some act that is not reasonable but bring it to the attention of the course's heads within 72 hours, the course may impose local sanctions that may include an unsatisfactory or failing grade for work submitted, but the course will not refer the matter for further disciplinary action except in cases of repeated acts.

Below are rules of thumb that (inexhaustively) characterize acts that the course considers reasonable and not reasonable. If in doubt as to whether some act is reasonable, do not commit it until you solicit and receive approval in writing from the course's heads. Acts considered not reasonable by the course are handled harshly. If the course refers some matter for disciplinary action and the outcome is punitive, the course reserves the right to impose local sanctions on top of that outcome that may include an unsatisfactory or failing grade for work submitted or for the course itself. The course ordinarily recommends exclusion (i.e., required withdrawal) from the course itself.

Reasonable.

- Communicating with classmates about problem sets' problems in English (or some other spoken language), and properly citing those discussions.
- Discussing the course's material with others in order to understand it better.
- Helping a classmate identify a bug in their code at office hours, elsewhere, or even online, as by viewing, compiling, or running their code after you have submitted that portion of the pset yourself.
- Incorporating a few lines of code that you find online or elsewhere into your own code, provided that those lines are not themselves solutions to assigned problems and that you cite the lines' origins.
- Reviewing past semesters' tests and quizzes and solutions thereto.
- Sending or showing code that you've written to someone, possibly a classmate, so that they might help you identify and fix a bug.
- Submitting the same or similar work to this course that you have submitted previously to this course, CS50 AP, or CS50x.
- Turning to the course's heads for help or receiving help from the course's heads during the quizzes or test.
- Turning to the web or elsewhere for instruction beyond the course's own, for references, and for solutions to technical difficulties, but not for outright solutions to problem set's problems or your own final project.
- Whiteboarding solutions to problem sets with others using diagrams or pseudocode but not actual code.
- Working with (and even paying) a tutor to help you with the course, provided the tutor does not do your work for you.

Not Reasonable.

- Accessing a solution to some problem prior to (re-)submitting your own.
- Accessing or attempting to access, without permission, an account not your own.
- Asking a classmate to see their solution to a problem set's problem before (re-)submitting your own.
- Discovering but failing to disclose to the course's heads bugs in the course's software that affect scores.
- Decompiling, deobfuscating, or disassembling the staff's solutions to problem sets.
- Failing to cite (as with comments) the origins of code or techniques that you discover outside of the course's own lessons and integrate into your own work, even while respecting this policy's other constraints.
- Giving or showing to a classmate a solution to a problem set's problem when it is they, and not you, who is struggling to solve it.
- Looking at another individual's work during the quizzes or test.
- Manipulating or attempting to manipulate scores artificially, as by exploiting bugs or formulas in the course's software.
- Paying or offering to pay an individual for work that you may submit as (part of) your own.
- Providing or making available solutions to problem sets to individuals who might take this course in the future.
- Searching for or soliciting outright solutions to problem sets online or elsewhere.
- Splitting a problem set's workload with another individual and combining your work.
- Submitting (after possibly modifying) the work of another individual beyond the few lines allowed herein.
- Submitting the same or similar work to this course that you have submitted or will submit to another.
- Submitting work to this course that you intend to use outside of the course (e.g., for a job) without prior approval from the course's heads.
- Turning to humans (besides the course's heads) for help or receiving help from humans (besides the course's heads) during the quizzes or test.
- Viewing another's solution to a problem set's problem and basing your own solution on it.

REFERENCES

[1] Alex Aiken. 2019. A System for Detecting Software Similarity. https://theory.stanford.edu/~aiken/moss/

[2] Harvard College Honor Council. 2019. Possible Responses. https://honorcouncil.fas.harvard.edu/possible-responses

[3] Timothy Paul Cronan, Jeffrey K. Mullins, and David E. Douglas. 2018. Further Understanding Factors that Explain Freshman Business Students' Academic Integrity Intention and Behavior: Plagiarism and Sharing Homework. *Journal of Business Ethics* 147, 1 (01 Jan 2018), 197–220. https://doi.org/10.1007/s10551-015-2988-3

[4] Guy J. Curtis, Bethanie Gouldthorp, Emma F. Thomas, Geraldine M. O'Brien, and Helen M. Correia. 2013. Online Academic-Integrity Mastery Training May Improve Students' Awareness of, and Attitudes Toward, Plagiarism. *Psychology Learning & Teaching* 12, 3 (2013), 282–289. https://doi.org/10.2304/plat.2013.12.3.282 arXiv:https://doi.org/10.2304/plat.2013.12.3.282

[5] Martin Dick, Judy Sheard, Cathy Bareiss, Janet Carter, Donald Joyce, Trevor Harding, and Cary Laxer. 2002. Addressing Student Cheating: Definitions and Solutions. In *Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education (ITiCSE-WGR '02)*. ACM, New York, NY, USA, 172–184. https://doi.org/10.1145/782941.783000

[6] Robert Fraser. 2014. Collaboration, Collusion and Plagiarism in Computer Science Coursework. *Informatics in Education* 13 (09 2014). https://doi.org/10.15388/infedu.2014.01

[7] Trevor Harding, Donald D. Carpenter, Susan M. Montgomery, and Nicholas Steneck. 2002. A comparison of the role of academic dishonesty policies of several colleges on the cheating behavior of engineering and pre-engineering students. *32nd Annual Frontiers in Education* 3 (2002), S1H–S1H.

[8] Amardeep Kahlon, Bonnie Mackellar, and Anastasia Kurdia. 2018. Combating the Wide Web of Plagiarism: the Discussion Continues (Abstract Only). 1069–1069. https://doi.org/10.1145/3159450.3162197

[9] Tony Mason, Ada Gavrilovska, and David A. Joyner. 2019. Collaboration Versus Cheating: Reducing Code Plagiarism in an Online MS Computer Science Program. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 1004–1010. https://doi.org/10.1145/3287324.3287443

[10] Christopher Moretti. 2016. Advanced Programming Techniques. https://www.cs.princeton.edu/courses/archive/spring16/cos333/

[11] Caleb Owens and Fiona A. White. 2013. A 5-year systematic strategy to reduce plagiarism among first-year psychology university students. *Australian Journal of Psychology* 65, 1 (2013), 14–21. https://doi.org/10.1111/ajpy.12005

[12] Vivek Pai. 2019. ETector. http://etector.org/

[13] Jonathan Pierce and Craig Zilles. 2017. Investigating Student Plagiarism Patterns and Correlations to Grades. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE '17)*. ACM, New York, NY, USA, 471–476. https://doi.org/10.1145/3017680.3017797

[14] Charles P. Riedesel, Alison L. Clear, Gerry W. Cross, Janet M. Hughes, Simon, and Henry M. Walker. 2012. Academic Integrity Policies in a Computing Education Context. In *Proceedings of the Final Reports on Innovation and Technology in Computer Science Education 2012 Working Groups (ITiCSE-WGR '12)*. ACM, New York, NY, USA, 1–15. https://doi.org/10.1145/2426636.2426638

[15] Simon, Judy Sheard, Michael Morgan, Andrew Petersen, Amber Settle, Jane Sinclair, Gerry Cross, and Charles Riedesel. 2016. Negotiating the Maze of Academic Integrity in Computing Education. In *Proceedings of the 2016 ITiCSE Working Group Reports (ITiCSE '16)*. ACM, New York, NY, USA, 57–80. https://doi.org/10.1145/3024906.3024910

[16] Stuart Zweben and Betsy Bizot. 2018. 2018 Taulbee Survey. Computing Research Association.