# An Exercise to Help Prevent CS139 Cheating

## Problem:

The students in CS139 are faced with larger programming projects than they may have seen in their high school courses or are dealing with large programs for the first time in this course. Due to time management issues and a fear of failing, some of these students feel compelled to break the rules and to work together or to outright steal code from one another. These cheating cases are stressful for the instructor as well as for the students.

## Opportunity:

Students may not realize that the impact of cheating and getting caught is far, far worse than not turning in an assignment at all. While some may not want to be complicit in a cheating case, peer pressure may force them into that situation. Providing them with some tools to rationally approach the problem may help to prevent further cases.

## Goal:

The goals of this exercise is:

* to make students aware of the kinds of cheating that can occur
* to help them rationally consider that the impact of a single failed assignment is negligible to their grade
* to provide them with some possible ways of preventing cheating if they are put into that position

## The Exercise

This exercise is done after the first (trivial) programming assignment, but sometime before the third (more difficult) assignment. It takes about ½ 50 minute class period with a preliminary reading assignment ahead of time. I have copies of the JMU honor code available as well as the individual worksheets.

### Method

Students read the 6 cases before class. (Sometimes for motivation I ask them to identify any that they think I have made up and which are actual incidents. In all cases, these are cases that I have found.)

In class, they are grouped together in teams of about 4 students. They each get one case with a series of questions. They are supposed to discuss with their group the answers to the questions.

The instructor randomly chooses teams to report back. In some cases, the entire class will get into the discussion. The instructor guides the discussion specifically asking questions such as “What is the worst that can happen if the student fails to turn in the assignment?” “When can the student go to talk to the professor?” etc.

### The Cases

**Panic Attack**

Frank was behind in his programming assignment. He approached Martin to see if he could get some help. But he was so far behind and so confused that Martin just gave him his code with the admonition that he could “look at it to get some ideas”.

In the paraphrased words of Frank: “I started the assignment 3 days after you put it up. But then other assignments came in and I started on them too. I felt like I was chasing rabbits and began to panic. It was already past the due date and I got really scared. That’s when I went to Martin to see if he could help.” Of course Frank copied the code and turned it in as his own.

**A Friend Indeed**

Jeffrey was having trouble with one of the last programming assignments. He didn’t even know where to begin and it was already late. Another student, Stephen, lived in his hall and he was pretty friendly with him. Jeffrey went to Stephen’s room and told him that his computer was acting flaky and could he borrow his to finish up the program? Stephen was on his way out to dinner and told him okay. When he got back Jeffrey was gone.

While Stephen was out, Jeffrey searched for and located his code for the assignment on his machine. Jeffrey copied it onto his floppy drive and took it back to his room, where he modified the code a bit before submitting it for a grade.

**Oops**

Emily was working in the lab on her programming assignment. She finally finished the program and submitted it and went on to do some other work. Shortly thereafter, she left the lab.

Another student, Kyle, was working nearby. He knew that she had successfully submitted the assignment, and he had not been able to get his to work properly. When Emily left, he noticed that she had not logged out of her computer. He moved to her workstation, found the work on her N: drive and copied it onto his memory stick. He then logged out, logged in as himself, and copied the code onto his N: drive where he modified the program a bit, then successfully submitted it.

**Too Close for Comfort**

Bill and Jeff were first year students and were rooming together in the fall semester and were taking CS139 together. Bill was an excellent student for whom the work in CS139 seemed to come easily. Jeff seemed to struggle a bit more, but was able to do the work and turned most labs and programs in on time.

Both programs came in looking virtually the same. When confronted, Jeff claimed the work as his own and stated that he did not know how the code from the roommates was the same. Later Bill told the professor that while programming was easy for him, he had struggled with another class and then got behind with this program. He didn’t think the professor would find out and so stole his roommate’s code and turned that in.

**Let’s Make a Deal**

It was mid-semester and the pressure was on, not only in CS139, but in other classes as well. Two students, Jamie and Pat were working on the programming test in the lab, but neither was having much success. Jamie had started the program days ago, but was having trouble debugging his current work. Pat had just started this day and knew that she would be late in turning it in. Pat offered to help Jamie work out the problems with her code. Together and after a couple of hours of work, they got the program to work. Pat said, “Now that yours is working, can you give me the code so that I can also get credit for this assignment.” When Jamie objected, Pat said, “Hey, you wouldn’t have gotten it finished if it weren’t for my help, and now mine will be even later.” So Jamie turned over a copy of the code. Pat made some changes to a few sections, and then turned the program in.

A friend of Pat’s also was having problems with the program. Pat offered the “community” code to her friend who changed a few variable names and turned in a third copy of the program.

**A Friendly Assist**

George was struggling with the latest programming assignment that was due to be submitted that night. He had gone home over the weekend, thinking that it would be easy to do this assignment, but it turned out to be more difficult than he thought. After working on some parts of it and giving up in frustration, he turned to Shelley an upper-class CS student who had taken 139 several semesters before. He showed Shelley the assignment and the two of them worked on it late into the night. It successfully submitted with only a one day late penalty.

## Sample of the questions

Which, if any, of the students were at fault? Why?

What JMU Honor Code violations occurred? – refer to the Honor Code sheet

What should Shelley have done in this situation?

What options did George have besides cheating?

# Time Management CS 239 Activity

## Problem:

The students have large programming projects to do in this class. The project is assigned and is due 2 weeks after the assigned date. My expectation is that overall they will spend from 10 – 20 hours on each project. (This is a four credit hour class and this is the main out of class homework with a lot of variability in individual student performance.) Students in CS139 have had similar although usually smaller projects throughout that semester. For this first project, many students were overwhelmed by the level of detail required by the assignment. They underestimated their time and as a result, the day it was due, many were panicking and demanding office hour time. Many turned the project in late or in an incomplete state.

## Opportunity:

This provided me with an opportunity to focus on their time management specifically related to the course. There was motivation due to the panicked state of many of the students.

## Goal:

The goals of this series of exercises were:

* to make the students more self-aware of how long the projects take
* to help them to see that there are more productive ways to spend that project time
* to reduce the level of frustration the last day that a project is due
* to help the students to do higher quality work of which they can be proud

## Transforming the Skill Bites exercise

My version looks quite a bit different from the Skill Bites original. Rather than focusing the students on how they spend all of their time (and having general non-compliance which I thought might happen) I wanted them to focus on how they spend their time related to that one activity. The categories for my exercise were: Design, Coding, Testing, and Debugging, 4 distinct tasks in the software development process. There is some “leakage” among the activities, but they are generally distinct phases. My goal is also to help them put more time into the Design phase so that they spend less time on the Debugging phase (which is where they burn many hours trying to fix code that does not work as intended.)

Steps:

1. Make the students more self-aware about their use of time on the last project while it was still fresh.
2. Prepare them to account for the time they spent on the project.
3. Provide some exercises that require that they spend additional time on the Design phase.
4. Continue to have the students monitor their time on task. (This will be ongoing all semester long.)

# Self Reflection and Sharing Exercise (This is Step 1)

# Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Survey about this first PA (Programming Assignment).

1. Did you submit the PA on time? \_\_\_\_\_\_\_\_\_\_ If you submitted to the test system, how many tests did you run? \_\_\_\_\_

If you did not submit to the test system, why not?

2. If yes, did you produce excellent work or did you just get it to the point of being able to turn in? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Estimate the total number of hours you spent on this project from when you first read the specifications to turning it in…if you are still working on it, estimate the number of hours that you have put in thus far. \_\_\_\_\_\_\_\_\_\_\_

4. Based on the total number of hours, how much time did you spend on?

* Design (thought process, building stubs, documenting, working through what each method might do)\_\_\_\_\_\_\_\_
* Coding (writing first pass code on the basis of the design that you used or wrote) \_\_\_\_\_\_\_\_\_\_\_
* Testing (running through tests to know what works and what methods have failed...include time spent running through submit since this is a set of tests as well). \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
* Debugging (after testing, correcting code and then retesting to correct a specific problem.)
* How much time prior to Tuesday? \_\_\_\_\_\_\_\_\_\_\_\_\_ Tuesday and Wednesday combined? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Did you unit test? (Test individual methods in Rainfall and RainfallIO in isolation to insure that they work correctly before trying to put them together? \_\_\_\_\_\_\_\_\_

7. Thinking about PA2, what one thing would you like to do differently from what you did in PA1?

8. What one thing in this PA gave you the most trouble? Be specific.

In your groups, share the results of the survey with one another.

Is there any difference among how the students spending less time spent that time? Is there anything you can learn from this? Describe

Looking at what gave you the most trouble, is there anything that your colleagues can suggest for next time? How did they go about solving that problem, or what resources did they use?

## Step 2: From the next Programming Assignment

### Timesheet

To help give you some idea of how you are spending your project time you will fill out a [timesheet.xls](file:///U:\Web\Courses\2008SPCS239\PAs\timesheet.xls), [timesheet.pdf](file:///U:\Web\Courses\2008SPCS239\PAs\timesheet.pdf), [timesheet.ods](file:///U:\Web\Courses\2008SPCS239\PAs\timesheet.ods) recording all of the time and the category into which that time goes for every block of time that you spend on the project. (Choose whichever format you prefer.  The pdf version is intended to be filled out by hand while the other two are intended to be electronic records.) While it may be scary to see how many hours you put into a programming project, this will help you to plan better for future projects in the class and others.  When working on customer work, your time might also be billed back directly to the customer necessitating an accurate accounting.  An explanation of the categories:

1. Design - Any time that you spend thinking through, charting, diagramming, or writing pseudo code is considered design time.  You may have a big chunk of design at the beginning, and then small amounts as you try to work through a particular problem or issue.
2. Coding - Any time spent writing code that is new to the project would be coding time.  Don't count the time that you are coding in direct response to an execution error from a test.  Do count as coding any time that you spend coding after doing a design or redesign and before you begin testing the new solution.
3. Testing - Any time spent in testing the program using data.  This might be unit testing (where you are simply testing a single component) or system testing (where you are testing how well the whole project works.)
4. Debugging - This is the time that you spend correcting and retesting problems that you find.  The testing and debugging process might lead you to return to design if a particular component is not working properly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Timesheet for : YOUR NAME GOES HERE** | | | |  |
|  | Indicate for each day how much time you spent on the various tasks.  Round to the nearest 10 minute interval. | | | |
| **Date** | **Design** | **Coding** | **Testing** | **Debugging** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Step 3: From the third assignment

### Continue having them keep a timesheet, but put an early design activity into the assignment.

Completed UML diagram for your application (your design) due in class in hardcopy form no later than Tuesday February 19th.  You may do this diagram by hand or if you prefer, there is Visual Paradigm installed on the lab computers which can make nice UML diagrams or you may download MS Visio (which is what I used to build the diagrams we have looked at) from the MSDN alliance for free.  [pdf format](file:///U:\\Web\\Courses\\2008SPCS239\\PAs\\Visio21Game.pdf) [visio format](file:///U:\Web\Courses\2008SPCS239\PAs\Visio21Game.vsd)

TIMESHEET - Again, keep a detailed timesheet.  Count as design any time that you spend on the UML diagram.  Also count as design any time you spend planning each method.  Make a conscious effort to put more time into design than you did on the last PA.  Timesheet is due stapled to the back of the program report on Tuesday.  Timesheets and their variants:  [Excel](file:///U:\Web\Courses\2008SPCS239\PAs\Timesheet.xls), [OpenOffice](file:///U:\Web\Courses\2008SPCS239\PAs\Timesheet.ods), [pdf](file:///U:\Web\Courses\2008SPCS239\PAs\Timesheet.pdf)

## Step 4 & Ongoing:

They will need to keep a timesheet for each PA. As we do each one, I will include a new challenge, such as putting 90% of the time in the week the assignment is assigned, or limiting the number of tests that they run through the final tester.

The original Skill Bites exercise follows.