

Gender-aware pedagogy for introductory CS classes

Daniel Bilar
Department of Computer Science
University of New Orleans
New Orleans, LA 70148
dbilar@uno.edu

ABSTRACT

TODO

Categories and Subject Descriptors

H.4 [TODO]: TODO; D.2.8 [TODO]: TODO—*complexity measures, performance measures*

General Terms

TODO

Keywords

TODO

1. INTRODUCTION

Trends in CS enrollment and graduation status quo overall[12], female[3] CS enrollment

”Women received about 38 percent of the computer science bachelor’s degrees awarded in the United States in 1985, the peak year, but in 2003, the figure was only about 28 percent, according to the National Science Foundation. At universities that also offer graduate degrees in computer science, only 17 percent of the field’s bachelor’s degrees in the 2003-4 academic year went to women, according to the Taulbee Survey, conducted annually by an organization for computer science research.”[4]

Ethical/fairness argument: Need to be inclusive of gender differences in teaching if true to calling of profession.

Utility argument: Need for the future CS graduates for competitiveness (NAS statement[13], data[5]) → cannot afford to vex 52% of population

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM JERIC '09

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

2. BACKGROUND RESEARCH

Anything empirical in science, math related to differences in gender learning styles and resulting need or experiences in adaptation of teaching methods. This will serve as a motivation for adapting CS curriculum

2.1 Learning styles

physiology learning behavior [14], science learning behavior[9], physics learning behavior (good solid paper!) [15] **need math and any CS learning style pointer**

2.2 Pedagogical Methods

math methods [1] **need two sciences, and any CS ped. methods**

3. A GENDER-AWARE APPROACH FOR COMPUTER SCIENCE

3.1 General Teaching Philosophy

Every course should have a high level philosophy, this is mine:

- Further intrinsic motivation (i.e. not through grades)[8, p.120]
- Mastery of learning, rather than performance goals[8, p.122]
- Student self-sufficiency (i.e teach them how to teach themselves)

3.2 Methods

In general, most syllaby are created ass-backwards: Instructors start with questions like “how many tests will I have, papers, grades” when the question should be: What are the *mastery goals* that I want them to achieve in this class, and then derive the lecture flow and evaluations towards those goals (it is more work than the other way round I will admit it). Once the instructor is clear in his/her mind what he wants them to learn, it has at least two great advantages: Methods can be adapted on the fly towards these benchmarks by taking student ‘learning pulse’ regularly and these benchmarks can be communicated to the student for self-checks.

3.2.1 Clear Expectations and Explanation on Course Goals on Syllabus

See my syllabus. Positive tone. Clear expectation, no lies and de-emphasis of grades and punitive messages. The

goal here is to convey the first in a series of messages that will further *intrinsic motivation* (i.e. the student wants to learn), not extrinsic motivation (i.e. if I do not learn I'll be punished with bad grades)

3.2.2 Open learning environment

Motto is that there are no stupid questions, mistakes are taken as chance to learn.

This is *vitally* important for women, since they are in general more sensitive to (mostly self administered) charges of 'stupidity'. Once students see that there is no penalty at all involved in asking questions learning can truly begin. Progress reports is the more private link in this chain.

3.2.3 Evaluation at all skills levels

Evaluations are made very often, at all levels except memorization of the learning pyramid of Anderson and Krathwohl [6], revising bloom (see table 1). No rote memorization is required (all is open book), because in my experience, one can learn symbols and reproduce them without any understanding of the underlying semantics. Intro to CS is about teaching mastery goals, not a memory exercise.

3.2.4 Structured Lectures

Old adage in public speaking states that you should say what you are going to say, say it, than say what you said. Effective teaching requires a similar approach. Students in general, but especially Type A overachieving women I encounter at these elite college, like structure, a list to 'check-off'. This entails sound class preparation for the instructor: A clear idea of the *learning goals* of each lecture which is written down (to practice) and written every class on the class board. A 50 minute class, one can be happy if 3-4 points come across clearly - 90 mionute class, maybe 4-6. Announce what you want them to get out of the lecture today - point it out as you go along - recapitulate including the comments made during class from the students.

A 50 minute lecture should require no more than a page of note, mainly as a conceptual scaffold for the lecture to come. See my class preparation notes.

3.2.5 Dynamic Class Notes

I encourage them not to take notes verbatim (concepts are fine) - I want them to pay attention and not be worried about missing a word. I write dynamic class notes every week and publish them online(takes about 2 hours), incorporating the questions and input and lessons from the class. See my class notes.

3.2.6 Socratic teaching methods

No one is going to be able to hide or learn passively in class. I constantly cold call people in roughly uniform order - everybody has to speak. For shy students, mostly women, this is how they learn to discuss, defend and articulate ideas, gain self-confidence in public speaking. In addition, reformulation ("active learning") forces them to reprocess in own words what they just heard - concepts become clearer.

3.2.7 Teach em how to fish

Learning strategies: Every discipline and course has their effective ways of achieving mastery of the material. CS is no different. Everyone, including the professor, runs into

walls. The main goal of any course is to teach them from the ground up how to behave towards the unknown, with respect, but not with fear - a form of panache that trusts, upon proper training, in one's critical abilities and methodological toolset to be able to tackle step by step the unknown. This self-trust has to come from experience, from active practice from earliest learning stages onwards

In CS this means periodically (once a week, on Mondays, while projects and labs are out and progressing etc) tapping into the collective class experience, guided and enriched by the experience of the teacher, in approaches to tackle these problems (design, debugging). Students will realize that they are not alone, and shyness in speaking up (characteristic of women in mixed classes, it seems) is meliorated. Example of syllabi teaching tips:

- Do the readings:
Read first the summary at the end of every chapter to get an overview of the chapter. Then read the chapter with a pen in hand before class, and mark questions, epiphanies, things that are unclear in the margins
- Experiment:
Do not be shy to go to the lab and write small programs for everyday tasks to try out concepts mentioned in class or in the book. This will get you familiar with logic errors, semantic error messages, the BlueJ environment and Java programming.
- Cooperative learning:
Explain the concepts in your own words to a fellow student, a TA, or to me, i.e. be a tutor to a tutee. This is so valuable I cannot stress it enough (one of the reasons we have progress reports). By doing so you consolidate and integrate your own knowledge, and you learn a great deal about how to learn since you have to diagnose the tutee's learning problem in order to help him/her overcome it.
- Debugging:
If you feel you are stuck debugging for an hour or two, stop. You have coded yourself into a corner and your brain is tired. Get up, get an ice cream, take a shower, walk around in the grass - anything to take your consciousness mind of it, for 15-30 minutes. Your subconsciousness will continue to work on it and when you return, you will find that you will be fresher in your approach and ideas.
- Start all projects early:
When you get stuck, you'll be glad you have the time to talk to me, the TAs, and other students to help you figure it out.

As the class progresses, more tips should be gathered from the students - they learn how to teach themselves (under guidance of course). The result is that a set of approaches will crystallize that serves as their toolset to tackle the unknown. As stated in the philosophy, the main goal of any course is to teach them from the ground up how to behave towards the unknown, with respect, but not with fear - a form of panache that trusts, upon proper training, in one's critical abilities and methodological toolset to be able to tackle step by step the unknown. This self-trust has to come from experience, from active practice from earliest learning stages onwards.

Table 1: Mapping of Evaluation Tools to Thinking Goals

Skill level	Description	Course Method
Higher Order Skills	Creativity: Reorganize elements into new pattern	presentation, projects
⋮	Evaluation: Give opinion and critique	labs, in class questions, presentation
⋮	Analysis: See and differentiate the parts of the whole	labs, in class questions, debugging
⋮	Application: Use heuristic toolset for problem solving	labs, in class questions, exercises, projects
⋮	Understanding: Interpret new concepts	progress reports, in class cold calling, questions, exercises
Lower Order Skills	Recall: Rote memorization	not used

3.2.8 Weekly Progress Reports

Weekly reports, one/two paragraphs to a page: In intro classes, every student sends this to the instructor at the end of the week, stating what they think they learned, what they found interesting, confusing, did not understand, finally did understand, ask questions about the materials, document the joys and tribulations and frustration of learning in a time series. Practically anything is permissible and I have seen practically every form: Lab report, dear diary, love letter, TPS (from the movie “Office Space”) report, chronicles, bug reports, apologies, poems and drawings. Some samples below from the same class week:

Elaine Benes

September 19-25, 2004

Sorry this is a day late, my computer has a virus and my e-mail is not working. This week in lab I learned a lot. I became more comfortable with the program and more confident in writing programs. I do feel that I have a grasp on what programming is about and how it works. On Friday, we learned about debugging and as usual when we learn new things in the class, I was very confused. I think that once I run more practice programs and spend some time actually debugging that I will be more confident with it. I have found that when something confuses me in class, that the concepts become much clearer when we run them in lab. The hands on time really helps give me a feel and a better sense of what we are doing. The analogies made in class and the examples we do on the board help to make abstract concepts more understandable. The lab is also interesting because it is interactive and it helps to make me want to learn more. I also feel a sense of accomplishment when I am faced with a problem and I am able to overcome it because I can actually see how I am improving and learning. Having pretty much no prior knowledge of what programming entailed it is nice to see and feel how I am improving because I have a greater appreciation for the class. I am also really inter-

ested by the subject and so that helps to ease the frustration because even when I am having trouble, I am still enjoying the time spent in the lab and learning. I think that with enough hands on time I will be able to understand more and pick up things more quickly. I’ve had a lot of fun so far, and hope that there will be more in the future.

You can discern the personalization which is more characteristic of female students. Next a male progress report for comparison’s sake:

James Bond

Sept 22 2005

This week we hit upon the key objectives of the class: abstraction and modularization. It is these modeling and thinking techniques that make for truly simple and intuitive programs. We looked at many different kinds of models, especially on their difference in focus compared to how the object or task was modeled. For instance, from the perspective of an FBI agent, the details about a person that are important are things like their name, age, family history, criminal record, finger prints, DNA structure; and if they were being modeled by a doctor, the doctor would look at them entirely differently. Although both cases model a human, the focus of the task is different, so some different properties of humans are modularized. The more time a project is spent in the thinking and planning phases, the less time and agony will be spent debugging and cursing.

Next we looked at the scope of a variable. This is an interesting feature implemented by Java that allows you to reuse names of variables and save on confusion and agony. Instance variables are created such that they may be accessed through any point in the class; but local variables and parameters are only recognized within the method they are created.

This leads to our next discovery: the “this” prefix. If you have a method that you wish to have

a parameter of the same name as a field (which happens often), then you use the fieldname for the parameter name, and set it to the instance variable, using the "this./instance variable"/ to differentiate between the "/parameter name" with the same name. /This is unimaginably useful and reduces the number of things one must remember outside of what the code intuitively tells you.

We then touched upon the difference between internal method calls and external method calls. Internal method calls invoke methods inside the same object, when external ones call upon other objects and tell them to execute their method of the same name. This is written "/Name of Object.Name of Method/" and is one way objects interact with each other.

We also ran into the topic of method overloading. This is another way of Java allowing us to make our programs simple and intuitive. We are allowed to reuse method names, along as they differ in number or their parameters. This is used if, sometimes you want to "CreateSquare" of set dementions, and othertimes "CreateSquare" and input the dimensions manually. Both of these constructors (for constructors are in themselves methods), would have the same name, but with different parameters.

I am enjoying the class and the lectures, and have no trouble with the tasks or anything else. Thank you and see you Monday!

The synopsis, analytical style is typical of most male progress reports. Progress reports by females serve to *externalize frustrations* lest the many roadblocks in intro course unduly impede learning. Women tend to internalize errors ("The computer says I am dumb because it says syntax error on the screen.") and periodic venting, in conjunction with an empathetic, encouraging ear by the lecturer, keeps motivation high.

Penelope Cruz CS151 Progress report September 23, 2005

I can't say that this week was easy. It is clear that the concepts we are talking about in class are getting more and more complex and difficult to understand. However, I can't say that I didn't expect this to happen. Monday and Wednesday I found the concepts about the keyword `this` and scope/lifetime of variables very confusing. This in turn made the lab more difficult. It seems to me that I can grasp most of the major ideas but somehow when put in front of the computer for a lab I don't know where or how to put them to use. The language used in the readings that we have to do I also find somewhat puzzling.

What helped me the most this week was today's lecture. I never knew that you had to look at the problem and outline as if it were a paper. I had figured that you just sort of took a stab at it. In any case, I feel pseudo-coding will help

me in future labs as well as the project coming up. After this week I'm still a little fuzzy on the dot notation. Is it only used when evoking another method or is there more to it that that? I also have a pretty clear understanding of internal methods calling other methods of the same class and external methods calling methods of other classes. I am also confident in understanding abstractions and modularization. (I must say the examples of books and maps were very helpful.)

Of all learning tools, I find progress reports is the most bang for the buck for the students and the instructor. The student gets a *sense of ownership of the learning process*, sees his/her progress, is forced at some level to grapple with class ideas at a regular basis and write down something about them.

The instructor has the finger on the class pulse, can gather questions, answer on the report or in class, give pointers to advanced students, encourage weaker ones, gauge class interest, so that the maximum number of students are and feel engaged in the class. In my experience, and I have been doing this for three years, it is an invaluable tool. Practical limit is probably a class with 30 students. It will take you about 3-4 hours to go through the reports on Sunday and answer questions, write down notes, etc. With this technique alone, I can in intro classes do in one class almost all the material that others do in two classes, and have 70%+ of students meet the major learning goals of the class.

3.3 Charisma

Bain has plenty of anecdotal evidence about Weber-esque^[7] charismatic leadership traits in the most successful teachers. Seduction and teaching, I find to be two sides of the same coin, you have to generate interest, keep excitement high, be playful, know how to give and take. In my experience, type A overachieving women will respect authority, not enforced by the baton (i.e. grades), but by charm, wit, intelligence and personal example. Many college women fantasize about male authority figures - this can be leveraged, delicately.

Let me make this clear: I am not advocating anything unethical, taking advantage sexually, leering, bullying, etc. Neither am I saying that one should conduct the classroom like a frat house, regularly make salacious comments or outrageous similes. I am simply stating that by leveraging charm, looks, zest, masculine energy, lots of humour and some seduction techniques used in different contexts for romantic pursuits, gives an extra nimbus to the instructor and serves to keep interest high. It is a fine line, and I do not recommend it for inexperienced men, slow-witted men or men who do not have enough psychological acumen re women to know how far one can go (not far, but you'd be surprised how much a little spice ever so often goes a long way - if only because students rarely encountered it before, if ever). This is not PC, but in my experience, it is supported.

4. RESULTS

Time period is two years (2004-2006), four classes of roughly 25-30 students, female makeup between 20% and 30%

4.1 Student evals

Discuss student evals, retention rates (if applicable) anecdotal - after two years of this, Colby had to double its intro CS class offerings - chair wrote to me offering as one

hypothesis my teaching (peer review from chair and other faculty can be quoted in support of this)

5. DISCUSSION

6. CONCLUSIONS

TODO The techniques and approaches mentioned above serves both the male and female student population. In my experience, four facets of intro CS need special emphasis for attracting and retaining female undergraduates and steering them towards the major:

- *Self-confidence buildup through ‘small-victories’ affirmation* is key: Like it or not, the first third of the semester the instructor is more of a psychologist, counselor, cheerleader. Once this hump has been taken, errors externalized, a feeling of ‘yeah it’s maddening but wow - I am able to do this’ has been established through successful completion of 3 or so labs, one project - it is smooth sailing: 4 out of 5 intro classes I taught since 2003 (my first was at oberlin), female students created the best final project of the class.
- An open, tolerant, playful, feedback rich, penalty free learning environment to ask question. This is very important - never ever tell shame a student for not getting a point. A positive spin (“you are 80% correct!”) works wonders.
- Assignments are tackled more enthusiastically when they have *social components*, not abstract solving (“write a program that draws a house adapted to the people’s gender, profession and age” ‘versus ‘write a program that sorts numbers”).
- Structured, ‘check-off’ lectures. I do not know why, but women love this format - every lecture imparts a sense of accomplishment and progress when one can check off what one understood (and the rest is asked in class and in teh progress reports)

One does not need to delve into much literature. Everything one needs is given re overarching goals in the seminal, but inexplicably marginalized “End of Education”[10]; the tried, true and tested techniques mentioned in McKeachie [8] and the periodic longitudinal studies/debriefings of the stellar masters of the field. As a very good, useful example of a deep study (15 years, over 100 college teachers) and resulting pedagogical excellence distillation, see Bain [2].

A word about teaching tools: I am generally sceptical re the latest and greatest electronic gizmos that are supposed to ‘revolutionize’ teaching, what postman termed the false “gods of Technology”. We have heard those promises before: Radio, to the Victrola, to 16mm film, CC TV, 8mm film, teacher-proof textbooks and today, computer-aided all around [10, p.50]. These tools, as trite as the observation may be, can only facilitate not replace good charismatic personal teaching, i.e they may make good teachers better [11], but they cannot make bad teachers any good, and may make them worse. Dedicated, prepared, enthusiastic teachers and proven teaching methods, as outlined above, calibrated and sensitive to the different learning styles (here gender) remain essential.

7. ACKNOWLEDGMENTS

This paper was made possible in part by the Norma Wilentz Hess Faculty and Program Fund in Computer Science at Wellesley College.

8. REFERENCES

- [1] O. A. Adedayo. Differential effectiveness by gender of instructional methods on achievement in mathematics at tertiary level. *Educational Studies in Mathematics*, 37(1):83–91, 1998.
- [2] K. Bain. *What the Best College Teachers Do*. Harvard University Press, April 2004.
- [3] S. Carlson. Wanted: Female computer-science students. *Chronicle of Higher Education*, 52(19):A35, January 2006.
- [4] C. Dean. Computer science takes steps to bring women to the fold. *New York Times*, April 2007. Accessed on August 18th, 2007.
- [5] P. M. Flynn. Skills challenges threatening u.s. competitiveness inthe global economy. In *National Skills Conference: The Skills Needs of the Irish Economy to 2020*. Expert Group on Future Skill Needs, October 2006.
- [6] D. R. Krathwohl. A revision of bloom’s taxonomy: An overview. *Theory Into Practice*, 41(4):212–218, 2002.
- [7] D. Ladkin. The Enchantment of the Charismatic Leader: Charisma Reconsidered as Aesthetic Encounter. *Leadership*, 2(2):165–179, 2006.
- [8] W. J. McKeachie. *McKeachie’s Teaching Tips: Strategies, Research And Theory for College And University Teachers*. Houghton Mifflin Company, 2005.
- [9] J. H. F. Meyer. Gender-group differences in the learning behaviour of entering first-year university students. *Higher Education*, 29(2):201–215, March 1995.
- [10] N. Postman. *The End of Education: Redefining the Value of School*. Vintage, 1996.
- [11] V. Razmov and R. Anderson. Pedagogical techniques supported by the use of student devices in teaching software engineering. *SIGCSE Bull.*, 38(1):344–348, 2006.
- [12] J. Vegso. Continued drop in cs bachelor’s degree production and enrollments as the number of new majors stabilizes. *Computing Research News*, 19(2), March 2007.
- [13] V. Vines. International scientists and engineers are essential for u.s. competitiveness. *National Academy of Science Press release*, May 2005. Accessed on August 18th, 2007.
- [14] E. A. Wehrwein, H. L. Lujan, and S. E. DiCarlo. Gender differences in learning style preferences among undergraduate physiology students. *Advan. Physiol. Edu.*, 31(2):153–157, 2007.
- [15] A. Zohar. Her physics, his physics: gender issues in israeli advanced placement physics classes. *International Journal of Science Education*, 25(2):245–268, 2003.

APPENDIX

maybe the syllabus, a lab, project description as an example? Maybe do not need an appendix