Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

Dan Garcia, Ph.D.
Lecturer SOE, EECS Dept, UC Berkeley
(on Sabbatical in Melbourne until 2008)

www.cs.berkeley.edu/~ddgarcia/
Overview

Quickie background
SIGCSE plug
This talk from 3 in 2007
ACE2008 in Wollongong
Improving CS education
  Teaching Tips
  Best Practices from survey of US schools
  ACM Education Board

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education
www.cs.berkeley.edu/~ddgarcia
Teaching Tips Archive

www.cs.berkeley.edu/~ddgarcia/teaching/tips/

Goal: Gather great CS teaching tips on a wiki

4 seasoned educators seeded with 10 tips each

Categories: Lecturing, Office (hrs), Staff, Exams, Labs, Section, Projects, Grading, Meta

Dan Garcia
UC Berkeley

Owen Astrachan
Duke Univ

Nick Parlante
Stanford Univ

Stuart Reges
U Washington

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

www.cs.berkeley.edu/~ddgarcia
Share Your CS Passion (Lecture)

For each of these, feel free to shout “Amen, Brother!”, and why it did / didn’t work for you

I show SIGGRAPH animations before every lecture

I talk about opportunities for students to join my graphics group (UCBUGG)

The videos are available for $40/video to ACM members

www.siggraph.org/publications/video-review/
ucbugg.berkeley.edu/

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

www.cs.berkeley.edu/~ddgarcia

SIGGRAPH 2007
FACE TOMORROW
Rather than having a single lecturer / TA cover every lecture / section, if two can attend each other's sections, pair up & alternate weeks.

ON week they do everything (TAs would cover both secs)
OFF week they sit in the ON's first section, as an observer, taking notes of how well the ON TA did, and debrief after

TAs Buddy TA in weeks 2 through n-1 (not first and last)
Learn tips from each other!
Can avoid stuff you hate doing (that co-inst may enjoy)

“Classroom Observation: The Observer as Collaborator”
–LuAnn Wilkerson

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

www.cs.berkeley.edu/~ddgarcia
We instituted them in our lower-div classes

Cost: ~$40/clicker

Two pricing models, subscription or purchase

Midway through lecture, ask “concept test”

Students vote w/clicker

2 min peer instruction time with neighbor
  – Must reach consensus

“Team” votes again

We discuss answers

Real-time feedback!

Can ask real-time Q!
Ask TAs teach a lecture
Have TAs and readers help write, beta-test, grade exams
Let TAs participate in syllabus discussions
Open your life to your staff
   Share cell numbers (great for emergencies)
   Grade exam at your home
   Take your admins out to a fancy lunch every term
Enlightened Grading (Grading)

Absolute grading, bump up at the end but never down
Allow later exams to replace earlier exam grade(s)
Give EPA! sprinkle points

- **Effort**
  - How much student tried
  - Office hours? Does all hw?

- **Participation**
  - Does the person ask Qs in lecture or discussion?

- **Altruism**
  - Helping others in lab / newsgroup / office hours

EPA grades are hidden, and can boost up ~ 1/2 ++ grade
Visible timer

Interrupt vs polling!

Show exam bug corrections on screen

…and can fix right there

Auto-announce time left

echo 'say "You may begin."' | /usr/bin/osascript
sleep 1800 # 1/2 hour
echo 'say "Sorry for the interruption... You have two hours left!"' | /usr/bin/osascript

… etc …

plaympeg ./epilogue.mp3
I presented 6 “hidden” tips from our collection

Lecturing
Office (hrs)
Staff
Exams
Labs
Section
Projects
Grading
Meta

As the Borg say:
Contribute your uniqueness
to the collective (our Wiki)
A Survey of Intro CS Curricula

Goal: Understand what top US PhD-granting schools did in their intro CS curricula

Survey sent out, collected, results presented

Today I’ll talk about the best practices we noticed
## Schools (2006 “US News & WR”)

<table>
<thead>
<tr>
<th>Rank</th>
<th>University</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CMU</td>
</tr>
<tr>
<td>2</td>
<td>MIT</td>
</tr>
<tr>
<td>3</td>
<td>Stanford</td>
</tr>
<tr>
<td>4</td>
<td>UC Berkeley</td>
</tr>
<tr>
<td>5</td>
<td>Cornell</td>
</tr>
<tr>
<td>6</td>
<td>UI Urbana-Champaign</td>
</tr>
<tr>
<td>7</td>
<td>U Washington</td>
</tr>
<tr>
<td>8</td>
<td>Princeton</td>
</tr>
<tr>
<td>9</td>
<td>UT Austin</td>
</tr>
<tr>
<td>10</td>
<td>U Wisconsin-Madison</td>
</tr>
<tr>
<td>11</td>
<td>Cal Tech</td>
</tr>
<tr>
<td>12</td>
<td>Georgia Tech</td>
</tr>
<tr>
<td>13</td>
<td>UC San Diego</td>
</tr>
<tr>
<td>14</td>
<td>U Maryland - College Park</td>
</tr>
<tr>
<td>15</td>
<td>Harvard</td>
</tr>
<tr>
<td>16</td>
<td>UCLA</td>
</tr>
<tr>
<td>17</td>
<td>U Michigan-Ann Arbor</td>
</tr>
<tr>
<td>18</td>
<td>Columbia</td>
</tr>
<tr>
<td>19</td>
<td>Purdue - West Lafayette</td>
</tr>
<tr>
<td>20</td>
<td>U Penn</td>
</tr>
<tr>
<td>21</td>
<td>Yale</td>
</tr>
<tr>
<td>22</td>
<td>Brown</td>
</tr>
<tr>
<td>23</td>
<td>Rice</td>
</tr>
<tr>
<td>24</td>
<td>UNC - Chapel Hill</td>
</tr>
<tr>
<td>25</td>
<td>Duke</td>
</tr>
<tr>
<td>26</td>
<td>U Mass - Amherst</td>
</tr>
<tr>
<td>27</td>
<td>USC</td>
</tr>
<tr>
<td>28</td>
<td>Johns Hopkins</td>
</tr>
<tr>
<td>29</td>
<td>NYU</td>
</tr>
<tr>
<td>30</td>
<td>Rutgers - New Brunswick</td>
</tr>
<tr>
<td>31</td>
<td>UC Irvine</td>
</tr>
<tr>
<td>32</td>
<td>Virginia</td>
</tr>
</tbody>
</table>

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

[www.cs.berkeley.edu/~ddgarcia]
What works really well and what is broken?

**Brown**
- High TA-student ratio
- Intro Experience Disparity

**Cornell**
- DrJava demos during lecture
- Closed required labs
- 1/2 Matlab, 1/2 Java

**Georgia Tech**
- (Compared to peers) very low attrition
  - 90% success in Media CS2
- No closed labs

**Harvard**
- Teacher training

**Princeton**
- Appl to commerce, science, engineering w/real datasets
- Too much to cover!
- Huge increase in enrollments

**Cal Tech**
- CS11 is several minicourses.
- Each student only takes 1 of these a term (of their choice)
- A course can be repeated three times for credit

**Cal**
- Closed required labs
- Undergrad teaching staff
- More pair programming

**CMU**
- All courses taught in small sections – no large lectures!
What works really well and what is broken?

Purdue
- Lecture, lab
- Hard to have creative projects but consistent grading

Rice
- Some want to move intro
- Scheme course to Java
  - Easier later transition
  - But, wide background variety!

Rutgers
- Online exams
- Java as intro language!

Stanford
- Undergrad section leaders
- Staffing in computer cluster
  - while students work on assignments, which are good
- We (all of us) still tuning CS1 Java

UC Irvine
- Informatics curriculum

UCLA
- Getting students to program to specs
- Intro experience disparity

UCSD
- Accelerated intro course track
- Closed labs
- Teaching memory allocation to beginners

UIUC
- All courses!
- Undergrad-led reviews
- Assembler EOS competition
- Programming studio structure
- Low-level understanding
- New student communities
What works really well and what is broken?

U Penn
- Compelling programming assignments

USC
- Intro experience disparity

UT Austin
- Students move well through CS2
- Java doesn’t allow them to learn memory management!
- One course is “geeked out” which could turn off non-programmers

UW
- Procedures-first intro to OOP

U Wisconsin
- Information overload
- Reduce sources of info
- Active and collaborative learning!

“Information overload has been mentioned repeatedly to me by students as a principle problem in many of their courses.

It is not uncommon for courses today to have a textbook or two, course notes, as well as lecture slides which overwhelm students with presentations of the same or similar material.

Add to that course email, web pages/links, blogs, etc. and students get trapped spending too much time reading and reviewing and insufficient time actively applying their newly gained knowledge.

What has worked well for us is to reduce the sources of information and to direct students’ efforts to more active and collaborative learning both in and out of class.”
What are the innovative techniques being deployed?

MIT

Overhauling EECS curriculum
Had been unchanged for nearly 30 years
6.001 (Scheme) out, Python in!
6.01 and 6.02 to be new mandatory intro courses
Offer intro to dept
Cut broadly across EECS
Strong hands-on component
Build robots, apply concepts like radio, Fourier transform, networks, large systems

Then choose 3 or 4 subjects from foundation courses
E.g., Circuits, Signals, Systems

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education
www.cs.berkeley.edu/~ddgarcia
What are the innovative techniques being deployed?

Georgia Tech

Computational Media degree!
Different entry points to major
CS1 for different majors
  Robotics
  Media Computation
Flexibility
  CS Majors can take any CS1

Contextualization
CS1s are typically dry!
Students today use flickr, youtube, etc -- leverage that!
Allow the data to be THEIRS
  movies, sounds, pix

Wil Justis, Milton HS

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education
www.cs.berkeley.edu/~ddgarcia
What are the innovative techniques being deployed?

UC Berkeley

UC-WISE
- Web-based learning system
- Interactive, collaborative & assessment
- Decrease passive learning in lecture (Old: 3 Lec, 2 Lab)
- Increase active learning in lab (New: 1 Lec, 6 Lab)

Top-down instruction
- Scheme, Java, C, Assembler

Self-paced courses
- 1-unit course for programmers in Matlab, Pascal, C, Scheme, Unix, C++, Java, Python
- Intro to programming in Scheme
- Awesome undergrad tutors!

www.ucwise.org

Teaching Tips, Best Practices, and Other Initiatives to Improve CS Education

www.cs.berkeley.edu/~ddgarcia
Conclusion, pt II

There is so much more analysis I didn’t present!

It’s all online; I encourage you to see our results.

There is some great innovation in the US!

MIT, GaTech, UC Berkeley, Caltech, UIUC, …

It would be great to learn about the innovation going on here to take back with me!

Survey!
ACM Education Activities

The ACM Education Board has three active task forces:

- Pipeline Crisis, Image
  *This was “enrollment”*
- Technology and Tools
- Curriculum

As a member of the Board, I want to learn what your problems are and how ACM could help!
One Take on Enrollment

- Our numbers roughly half that of the 2000 peak
Numbers across US half that of 2000
Some coming back
Need for computing professionals growing
Computing image poor
Nerds, no life, just coding
Computing Degrees & Careers site launched
Supplemental brochure sent to all US high schools
Problem
Computing educators (newbies and vets) don’t know of all available classroom technology and tools (too hard to keep up)

Solution (still in beta)
Build site for them
Just links w/comments
What is missing?
Support Web 2.0
Ratings, tags, how it worked for you, etc.
Take-away messages

Teaching tips
- What are yours?
- Contribute to our wiki

Best practices
- What are yours?
- Read survey for more

ACM initiatives
- How could ACM help?
- Talk to me about issues!

Let’s collaborate!