Engaging Students in Active Learning of Computer Science Concepts

Susan H. Rodger
Duke University

Feb. 3, 2015
ACM–iSIGCSE Workshop
Pune, India

Supported by NSF Grants DRL-1031351, DUE 1044191, CRA, and IBM Faculty Awards

Thanks for inviting me!

• Cold enough to snow back home!

or ?

• Tell you about SIGCSE from my experiences

Back in 1989

• PhD at Purdue University

• Assistant Professor at Rensselaer Polytechnic Institute

• First Task – combine first two computer science courses with automata theory

Back in 1990

• Tool – NPDA - to experiment with pushdown automata
SIGCSE

• 1990 – interested in CS Education
• 1992 – my first SIGCSE conference
• 1996 – my first ITiCSE conference
• 2006 – my first ICER workshop
• 2015 – my first ACM India conference!
• 2015 – my first iSIGCSE event!

Never missed a SIGCSE Symposium since I started going

• 1997

First son

2000

Second son

Brought help

What SIGCSE has meant to me

• Great colleagues
  • Friendliest people
  • Friendliest conferences
• Learn about innovative pedagogy
  • Use in my courses
• Share
  • My ideas
  • My curriculum materials and software

Outline

• Lecture formats
• Prepare for Class
• Class participation
• Engaging with Software
• Engaging in other ways
Motivation

• Traditional way of teaching
  • Professor Lectures
  • What students hear about 87%
    BLAH BLAH BLAH BLAH BLAH

Active Learning - Workshop Format
 "Flipped" Classroom

• Lecture for 10-20 minutes
• Students work on problem with computers in pairs
• Bring students back together
• Pair programming

Alternative

• Work in pairs
• Everyone has their own laptop

Groups/Pairs

• Assigned

CompSci 4 Section 1
Pairs as of October 22, 2009

Front of room
G1 G2 G3 G4
G5 G6 G7 G8
G9 G10 G11 G12 G13 G14 G15
G16 G17 G18 G19 G20
G21 G22 G23 G24 G25

Group 1
Starkweather, Clara cks15@duke.edu
Dinkins, Tiffany tiffany.dinkins@duke.edu

Group 2
Listenbee, Kamaria kkl7@duke.edu
Brown, Dwayne dcb26@duke.edu

Group 3


Outline

• Lecture formats
• Prepare for Class
• Class participation
• Engaging with Software
• Engaging in other ways

Read the book

• Read before coming to class
  • Ready to work in class
• Reality
  • Run out of time to read, not prepared
• Bring on – Reading quizzes
  • Online
  • Turn off when class starts

Have an engaging book....

• Runescape (Brad Miller)

Electronic Textbooks (ebooks) engage students

• OpenDSA (Shaffer, Virginia Tech)
  • Algorithm animations built in
• runestoneinteractive.org (Brad Miller,
  • Several books (Python)
    • Python - try and run code built in
    • Quizzes
• Zyante.com – interactive textbooks
• Track student progress
• Requirements and design strategies for open source interactive computer science eBooks
  • ITiCSE 2013 Working Group (Korhonen, Naps, et al)
Use engaging and visual tools
Python Tutor

```
1 scores = [10, 8, 3, 9]
2 list2 = scores
3 list3 = scores[:]
4 scores[2]=5
```

Problem Solving with Feedback

• APT – Algorithmic Problem Tester
• Test one function
• Runs on multiple inputs

Outline

• Lecture formats
• Prepare for Class
• Class participation
• Engaging with Software
• Engaging in other ways

Ways to Select students to answer questions

• Problem – same students always eager
• How do you get other students to participate?
  • Randomly call on them
  • Pick A Student program
  • Work in groups – call on group
  • Assigned groups – call on group numbers
Randomly Select a Student
Pick A Student Program

Collect pictures of students
program that cycles through and randomly picks one
Remove, then start again

From Owen Astrachan

Google Forms

Mystery While
* Required

NETID of person 1 *
Example: abc123

NETID of person 2
Example: abc123

NETID of person 3
Example: abc123

NETID of person 4
Example: abc123

Names of people filling out form *
(first and last name for each person, separate each name by a comma)

Google Forms (cont)

What does Mystery2 do (in words)?

What does Mystery3 do?*
(in words)

Google Forms (responses)

<table>
<thead>
<tr>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>What does Mystery3 do?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>If counts every character in a word except for lowercase 'e's</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It counts the number of characters in the word that aren't lowercase 'e's</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is counting the number of characters in the word that are not 'e's</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Count the number of characters that are not 'e' in the word</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mystery 3 returns a given a given word without the lowercase 'e's</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counts the lowercase 'e's in the word</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It returns the number of characters in a word that are not e</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counts all of the letters in word that aren't 'e'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counts the number of characters that are not 'e' in the word</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counts all the characters that aren't 'e'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counts how many letters there are that are not 'e'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mystery 3 counts the characters in a string that are not 'e', then returns the total count.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Setting up Google Forms

• Make it easy for students to get form

Outline

• Lecture formats
• Prepare for Class
• Class participation
• Engaging with Software
  • JFLAP
  • ALICE
• Engaging in other ways

Learner Engagement Taxonomy with visualization software

• Different forms of Learner engagement
  • No Viewing
  • Viewing
  • Responding
  • Changing
  • Constructing
  • Presenting

• ITiCSE Working Group Report 2002 (Naps et al.)

Formal Languages and Automata Theory

• Traditionally taught
  • Pencil and paper exercises
  • No immediate feedback

  • More mathematical than most CS courses
  • Less hands-on than most CS courses
  • No programming? Unlike most other CS courses
## Why Develop Tools for Automata?

<table>
<thead>
<tr>
<th>Textual</th>
<th>( {(q_0, q_1, q_2), {a, b}, \delta, q_0, {q_2, q_0}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \delta = {(q_0, b, q_0), (q_0, a, q_1), (q_1, a, q_0), (q_0, b, q_0), (q_2, a, q_0)} )</td>
</tr>
<tr>
<td>Tabular</td>
<td>![Tabular Diagram]</td>
</tr>
<tr>
<td>Visual</td>
<td>![Visual Diagram]</td>
</tr>
<tr>
<td>Interactive</td>
<td>![Interactive Diagram]</td>
</tr>
</tbody>
</table>

## Overview of JFLAP

- **Java Formal Languages and Automata Package**
- Instructional tool to learn concepts of Formal Languages and Automata Theory
- **Topics:**
  - Regular Languages
  - Context-Free Languages
  - Recursively Enumerable Languages
  - Lsystems
- **With JFLAP your creations come to life!**

## Thanks to Students - Worked on JFLAP and Automata Theory Tools

- NPDA - 1990, C++, Dan Caugherty
- JFLAP - 1996-1999, Java version
  - Eric Gramond, Ted Hung, Magda and Octavian Procopiuc
- Pâté, JeLLRap, Lsys
  - Anna Bilska, Jason Salemme, Lenore Ramm, Alex Karweit, Robyn Geer
- JFLAP 4.0 – 2003, Thomas Finley, Ryan Cavalcante
- JFLAP 6.0 – 2005-2008 Stephen Reading, Bart Bressler, Jinghui Lim, Chris Morgan, Jason Lee
- JFLAP 7.0 - 2009 Henry Qin, Jonathan Su
- JFLAP 8.0? – 2011-14 Julian Genkins, Ian McMahon, Peggy Li, Lawrence Lin, John Godbey

## DFA Example

- Build a deterministic finite automaton (DFA) to recognize **even binary numbers** with an **even number of 1s**.
- Only use symbols 0 and 1
- Binary numbers: 0, 1, 10, 11, 100, 101, 110, 111, ...
- When is a binary number an **even number**?
  - Ends in 0
- Which strings should be accepted?
  - 11010, 10010, 1111, 10100

| No, odd no. of 1's | Yes | No, ends in 1 | Yes |
Accepts Input! 1101010

Add meaning to states!

- Only one 0
- Odd number of 1's
- Odd number of 1's, ends in 1
- Even number of 1's, ends in 1
Alice Programming Language

• Create interactive stories or games
• Learn programming in an easy way, drag-and-drop your code
• Problem solving with visual feedback
  • Logical thinking, Computational thinking
• Along the way, learn computer science concepts:
  • Loops, classes, methods, functions, arrays

Alice Developed by Randy Pausch

• Carnegie Mellon University
• Virtual Reality Researcher
• Wrote the Last Lecture
• Died of Pancreatic Cancer in 2008
More on ... Alice Programming Language

- Has libraries of 3D objects
- Keeps Track of objects you select

Objects Have Multiple Parts that are moveable

Object Position

- Objects
  - Are positioned in 3D space
  - Have six degrees of freedom

Alice Code is Easy to Learn

Select Code, Drag-and-Drop code in program
Play Alice Animation

• Chicken rises, cow turns head and talks

Adventures in Alice Programming

• 2-week Teacher workshops
  • Over 200 teachers, middle school, high school, some elementary
  • First week Teach Alice, Practice
  • Second week - Develop Lesson Plans
  • One-week follow-up workshop the following summer
  • Summers 2008-2015, funding for lodging

• Main Sites:
  • Duke University, Durham, NC
  • Charleston/Columbia, SC
  • San Jose, CA (starting 2014)

Science Example

How a volcano is formed
Math Story on Fractions

Curriculum Materials

• Over 90 tutorials available for free
• Beginner, advanced, challenges, projects
• Paper handouts and video
• Teacher lesson plans in many disciplines

Example: Getting Started Tutorial teaches:

• Placing objects
• Moving objects
• Setting up Camera tripods and moving between views
• Using built in methods and writing your own
• Gluing objects together
• Adding sound, 2D pictures to enhance world

Getting Started Tutorial – 3 part
Sample tutorial: Scene Change

New Tutorial – Camera views following a person

Tutorial for Simple Game – Control boat, earn points

Fun with Alice
What a 6th grader can do with Alice - teacher Chari Distler

No Superheros in Alice

Outline

- Lecture formats
- Prepare for Class
- Class participation
- Engaging with Software
- Engaging in other ways

Example: Be a Robot

- 4 People
  - Controller (head)
  - Sensors (eyes)
  - Manipulators (2 hands)
- Blindfolded except eyes
- Controller knows what to build
- Limited communication

SIGCSE 96, Rodger, Walker
CS Concepts Coming Alive - Back in 1989

• New Assistant Professor at Rensselaer

• What data structure is this?

Engaging students in a group activities/large course

• Problem Solving in groups
  • Clickers, Google forms – compare results
  • Flip Classroom, reading quizzes (turn off at start of class)

• Acting out stories, games
  • Everything I needed to know about teaching... - Pollard, Duvall (SIGCSE 2007)

• Acting out algorithms with the whole class
  • Make a binary tree with the whole class
  • Calculate the height of the tree
  • Making Lemonade … large lecture classes – Wolfman (SIGCSE 2002)

• Acting out algorithms with a subset of students
  • Sorting algorithms – selection sort, insertionsort, etc
  • CS Unplugged activities

Middle School students sorting themselves with Bubblesort

Card Class – shuffling, dealing hands

• Poker hands – Full house, Flush, etc.

Cards
Royal Flush

Notable Women in Computing Cards
bit.ly/NotableW

- Based on Wikipedia project – wrote guide on how to write a Wikipedia page on a Notable women in Computing
- Picked 54 Women - deck of cards
- Page on using cards to teach CS
- Poster of the women

3 female
Turing Award
Winners

6 women
Eniac Programmers
Anita Borg
Grace Hopper
ACM Fellows
Other

Interaction in Class – Props
Passing “Parameters” in Class

- Pass by reference – throw frisbee
- Pass by value – throw copy of frisbee
- Pass by const reference – throw “protected” frisbee
Interaction in Class – Props
Linked List and Memory Heaps

Interaction in Class – Props
Memory Heap

ITICSE 98 – Astrachan – “Concrete Teaching: Hooks and Props as Instructional Technology”

The edible way to engage students
About Me - Hobby – Baking Shape cakes

How do you make those cakes?

What happens when your hobby and your career collide?

It is now time for engaging students with edible CS

CS 1 Sorting Cookies
Automata Theory
Interaction in Class – Props
Edible Turing Machine

• TM for \( f(x) = 2x \) where \( x \) is unary

• TM is not correct, can you fix it? Then eat it!

• States are blueberry muffins

Students building DFA with cookies and icing

Alice Programming Language

Conclusions
• We have shown several ways to engage students with CS concepts and make those concepts come alive
• Questions?