AP CS Principles Pilot at University of North Carolina at Charlotte

**Author:** Tiffany Barns  

**Course Name:** The Beauty and Joy of Computing (course number pending)  

### a) How BJC Fits In At UNC Charlotte:

UNC Charlotte’s Computer Science department offers ITCS1212, *Introduction to Computer Science*, a course with an accompanying lab, which is taught to 300 students (typically freshmen) from across campus each semester. It is a required course for a few departments. ITCS1212 is consistently over-subscribed and further expansion is resource limited. This course is the first in a series of courses for CS and IT majors, and teaches basic programming concepts in C++. Non-majors find this course difficult and leave it unsure about how it connects to any other subjects.

ITCS 1203 *Beauty and Joy of Computing* is a preliminary course, cross-listed for Computer Science and Information Science, intended to introduce non-majors and majors with little to no computing experience to the general concepts of computing. Beauty and Joy of Computing stresses the connection between computing and other disciplines while building a strong foundation in general computing concepts through hands-on visual and interactive projects.

### b) Course Stats

- **Lectures:** 2 75-minute periods, TTh  
- **Labs:** Built into the lecture schedule, about ½ of class time  
- **Course:** 16-week semester, yielding 40 contact hours  
- **Credit Hours:** 3  
- **Fulfills Requirements:**  
  - Computer Science General Elective (Spring 2011/Fall 2011)  
  - **Future:** Hopefully General Ed, Mathematical and Logical Reasoning  
- **Attendance:** 22 started; 20 finished  
- **Programming Language:** SNAP! (BYOB, a version of Scratch), AppInventor, GameMaker, and StarLogo TNG  
- **Grading:** 21 homework assignments, 13 labs, midterm, final  
- **Pilot:** Spring semester, 2011

### c) Class Content

The class followed the schedule shown below.

Classes usually consisted of an introduction to a topic via Powerpoint slides and videos, a CS Unplugged style class activity or a discussion, and a lab. Class topics were chosen to be relevant and to illustrate computing’s connection to society and innovation. Labs gave students hands-on experience with the material. Most of the lab exercises were done in pairs to be completed as homework. Students were thus
motivated to complete as much of the labs in class as possible. The intention was for each lab to be doable by students in under 2 additional hours outside of class. Weekly readings emphasized the impacts of computing, both positive and negative, on society, and were usually discussed through online forum discussions.

The class focused on making computing relevant while also demonstrating fundamental computing concepts in a hands-on interactive way. Guest lectures from departmental faculty and a doctoral student emphasized computing research topics and how they impact society.

• Richard Souvenir – Computer Vision (classifying what’s in a picture/video)
• Jamie Payton – Participatory sensing (incentivizing crowdsourced data collection)
• Lane Harrison – Visualization (how it can help people solve problems)
• IBM Watson – Campus-wide presentation by a Project manager at IBM
• UNC Charlotte research overview; opportunities for undergraduate research

Students reported that the class filled gaps they had in computing knowledge, even for those who took the course as an elective in their senior year in computing. Two students applied for and participated in a summer REU after the course; one of these was a business major who is changing his major to IT.

Topic List for UNC Charlotte’s Beauty and Joy of Computing course

1. Welcome & Orientation
2. Scratch Intro lab
3. Scratch Controls & Variables lab
4. Video Games lecture
5. Gamemaker Intro lab
6. BYOB – make your own blocks lab
7. OO Programming; Controls lab
8. Algorithms lecture; Lists Intro lab
9. Complexity; Using Lists lab
10. Artificial Intelligence - Videos
11. Search and Sort Algorithms
12. Recursion Unplugged activity
13. Recursion; Fractals lab
14. Midterm review
15. Midterm
16. Project Introduction
17. Project Proposal Presentations
18. Applications that Changed the world – demos; discussion
19. Computer Vision & Visualization guest lectures
20. Research opportunities & Visualization guest lecture
21. HTML and usability
22. Project work (lab)
23. Usability testing of projects
24. Project presentations
25. AppInventor lab
26. AppInventor lab 2
27. Simulation BYOB lab
28. StarLogo TNG simulation lab
29. Higher Order Functions lab
30. Final exam (includes societal impact essay)

d) **Evidence of Student Work** Of particular importance in our class is promoting students to use computing for their own goals. Projects ranged from a math multiplication game, to a Jeopardy! Interface, to a colonial times history lesson, to a
tournament scheduler, to a memory and math game that was actually used for kindergarteners.

Water Bender was a favorite project in the class. It’s a math practice game, where you solve randomly generated multiplication problems to save a tree; solving one problem douses one spark that can ignite the tree. This program was quite challenging for the students to design – the sparks travel toward the tree, and speed up as time goes on, becoming faster as the player misses more problems. The impressive thing about this game was that it was tuned to match with the player’s ability. This is difficult to accomplish in any game, let alone one made in 4 weeks by novices.

e) **What Worked And Didn’t**

*Scratch/BYOB/ SNAP!* This easy-entry programming environment based on one made for children made it easy for novices to learn to program. However, we did jump right in and scared a few might-have-taken-this-class students away with labs to complete at home. The extension of Scratch that allows you to make your own blocks, and those that allow for higher-order functions, make it easy to teach advanced concepts in an early course.

*GameMaker, AppInventor.* Students were very excited to learn to make games and Android phone applications. GameMaker was easier for programming large numbers of objects, and made a great contrast for talking about programming
paradigms, since it is more strongly object-oriented than Scratch/BYOB/SNAP! App Inventor helped students feel that they could make cutting edge programs that other people might actually use. Since it uses the Scratch programming interface, learning it was easy.

*StarLogo TNG:* This environment felt clunky compared to the rest, and since it ran slowly on our old laptops, this feeling was magnified. Some students liked the simulation aspect but we felt we could have found or created labs that made a better connection to biology and student’s lives or at least other subjects.

*HTML.* We did one class and homework on HTML but it was much less exciting than other topics. Many students had already looked at HTML before; we felt we could have skipped this topic since students can learn it easily on their own.

*Abstraction.* It’s the #2 idea in the “Big Ideas” list, and it was a topic addressed most weeks. We felt this was a success – students did seem to be working at the right level of abstraction on most projects. This might not be as explicit as it could be.

*Sequencing.* Student feedback showed appreciation for learning multiple tools but wished for them to be “chunked” so all BYOB was together and the other applications came later. This was done in Fall 2011 and worked well.

*Projects.* Allowing students to choose their own projects and teams was a great success. We initially planned for 2 projects but reduced it to one after we saw the high quality of those turned in. In Fall 2011, we went back to 2 projects but the second was a small one to demonstrate deeper understanding of one newly introduced tool (GameMaker, AppInventor, or StarLogo TNG). This allowed students to spend more time in one tool they enjoyed. Usability testing in class really drove home the importance of testing code and just how difficult it is to get that final level of polish to a piece of software.

*Readings.* Understanding of the readings would be better if they were explicitly discussed in class or if there were peer reviews of one another’s posts in the forums.

f) *Links, Resources, Acknowledgments*

We gratefully acknowledge support for this pilot from the College Board, funded by grant 0938336 from the National Science Foundation.

- The class Moodle (login as guest): [http://moodle.game2learn.com/](http://moodle.game2learn.com/)
- The BYOB/SNAP! Page [http://byob.berkeley.edu/](http://byob.berkeley.edu/)
- GameMaker: free download available at [http://yoyogames.com](http://yoyogames.com)
- App Inventor: [http://appinventoredu.mit.edu/](http://appinventoredu.mit.edu/)
- The Beauty and Joy of Computing “Frabjous CS” project site for Dan Garcia, Brian Harvey, and Tiffany Barnes project to train 100 HS teachers to teach BJC: [http://bjc.berkeley.edu](http://bjc.berkeley.edu)
UNC Charlotte’s Beauty and Joy of Computing class was adapted from Dan Garcia and Brian Harvey’s BJC class at Berkeley. The UNC Charlotte class is distinct in that it has less than half of the contact hours of the Berkeley course, but supplements the course with learning multiple programming environments and tools. This is accomplished via fewer and shorter lectures, lab time in class, and discussion of readings done online.

The pilot teaching assistants were Drew Hicks (then and now a doctoral student and now NSF Graduate Research Fellow) and Shaun Pickford (then undergraduate, now graduated with a job at Microsoft). Drew and Shaun helped extensively in preparing course lectures and CS Unplugged activities.

Four high school teachers in Charlotte including Sharon Jones, Beth Frierson, Renada Poteat, and Brian Nivens, observed some days of the course, worked through labs, and designed assessment questions for adoption when the course is offered in high schools.

© ACM, 2012. This is the author’s version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in ACM Inroads, {VOL 3, ISS 2, (June 2012)} http://doi.acm.org/10.1145/10.1145/2189835.2189855