Human Language Learning as a Metaphor for Teaching CS1-2

Edward L. Jones
Florida A&M University

Outline

- Challenges of Teaching CS1-CS2
- Trends - Student Engagement
- Sage Advice Revisited
- Human Language Learning Metaphor
- An Example
- What Next?
- Acknowledgments
CS1- CS2 Challenges

- Negative indicators
  - High failure rates (up to 65%)
  - Poor mastery
  - Undesired behavior -- plagiarism
- Under the hood …
  - Lecture based
  - Program-construction dominant activity
  - Individual mastery vs collaboration
- An even closer look
  - Cognitive jungle (Bloom’s taxonomy)
  - So many “languages” (skills) to learn

Trends: Student Engagement

- FUN!! - robotics, gaming, multimedia …
- Active Learning
  - Increased interaction, facilitated learning
  - Reduced role of lectures
- Studio-Based Learning
  - Collaborative problem solving
  - Feedback / reflection
- Costs
  - Technology support
  - Behavior changes
Sage Advice (Mary Shaw, 1991)

- **MORE** of this:
  - Study examples of good software
  - Exercises to modify / combine programs
  - Experience using existing software solutions

- **LESS** of this:
  - Programming from scratch
  - Assignment of throw-away programs
  - Emphasis on abstract skills at the expense of concrete experience

---

### Learning a Foreign Language

- **What's the fastest way to learn a language?**
- **ACT LIKE A BABY.**

- More than 30 languages available.

---
Three characteristics of the child’s language-learning process are crucial for success:

First, and most importantly, a child’s natural language-learning ability emerges only in a speech-soaked, immersion environment free of translations and explanations of grammar.

Second, a child’s language learning is dramatically accelerated by constant feedback from family and friends. Positive correction and persistent reinforcement nurture the child’s language and language skills into full communicative expression.

Third, children learn through play, whether it’s the arm-waving balancing act that announces their first step or the spluttering preamble to their first words. All the conversational chatter skittering through young children’s play with parents and playmates—“…what’s this…” “…clap, clap your hands…” “…my ball…”—helps children develop language skills that connect them to the world.

Language(s) learned via immersions

- Speaking ➔ Reading ➔ Writing
- Social interaction is key (AL/SBL)
- Outcomes include critical thinking and communication

Software is multi-lingual

- Requirements, specification, design, code, application
- Code is just one of the many languages

HLLM ➔ Macro design of holistic learning experiences
## Sample Holistic Assignment

### Charter
Understand intent; Code from design.

### Inputs:
(1) Need statement; (2) Flowchart; (2) Solution key (executable only);

### Solution stages:

| WHAT | Goal: understand/verify design.  
Create problem instances from Need; 
Execute solution key to verify instances; 
Critique/discuss discrepancies. |
|------|--------------------------------------------------------------------------------|
| HOW  | Goal: retain artifacts for future use. Submit all artifacts.  
Code the solution design; 
Verify code compliance to design; 
Reflect/discuss required corrections. |
| BUILD| Execute code on problem instances. 
Reflect/discuss discrepancies in code design, need statement. |
| VERIFY|                                                                                     |
| SUBMIT|                                                                                     |
Some Driving Principles

- #0: Programming is multilingual
- #1: Consume before producing
- #2: Drill for skill
- #3: No answer is ever really wrong – Teachable moments (reuse artifacts)
- #4: Invest in infrastructure

Programming is Multi-Lingual

- Technical specifications varying in formality of syntax and precision of semantics
  - Problem specification (intent)
  - Design specification (guidance)
  - Code (unforgiving syntax, semantics)

- Translation across specifications
Consume Before Producing

- Ability to read a language should precede a demand to write in it
- Higher-order language constrains; lower-order language elaborates
  - Write code based on a design
  - Given a specification, critique design
  - Given requirements, critique candidate solutions

Drill for Skill

- Mastery of basic skills a prerequisite to participating in active learning
- Lecture no substitute for dirty hands!
- Require coding/debugging drills on selected construct(s)
- Drills must not be optional
  - Count significantly in grade
  - Instrumented drill environment
Seize Teachable Moments

- Making mistakes is inherent in programming
- Recognizing and fixing mistakes develops critical thinking skills and provides opportunity for interactive group learning
- Students refine/verbalize their “value systems” for good/correct software

Infrastructure for Change

- INERTIA: No extra effort/resources needed for “business as usual”
- Significant infrastructure needed to support HLLM
  - Multilingual software artifacts
  - Instrumented drill environments
  - Observational environment (behavior, intervention)
  - Holistic assignment templates
  - Army of TAs!!
What Next?  For CS1 …

- Create HLLM lesson plan templates
- Create artifacts repository
- Acquire/develop tools for drills
- Elaborate lesson plan templates as individual & group assignments
- Assess costs and impacts

Acknowledgments

- This work is supported by NSF CPATH Award CNS-0939138 - CT-Driven Evolution to Studio-Based Active Learning
Questions??