Closing the Loop on Personalized Programming Learning in Blended Instruction Classes

Dr. Sharon Hsiao
Arizona State University

Sharon.Hsiao@asu.edu
http://www.public.asu.edu/~ihsiao1/
What have I been doing?

- **CSI (Computing Systems & Informatics) Lab**
  - [https://sites.google.com/a/asu.edu/csi/](https://sites.google.com/a/asu.edu/csi/)
  - 1 PhD (I just admit 2 more 2017 Spring)
  - 3~5 MS (Graduated 3)
  - 3 Undergraduates

- **Community:** LAK, ITS/AIED, EDM, CSCL, ICALT, ECTEL
  - IJAIED in CS associate editor
  - 2017 AIED & EDM social media chair
Experience the Joy of Programming
Research in Computer Science Education

1. Problem Solving
   – Parameterized quizzes
   – Multiple choices (Quiz Of the Day)

2. Visual Learning Analytics (≈OSM)
   – Semantic topic modeling + visualization (LAK’15 & ’16)

3. Behavior/Learning Activity Modeling
   – Programming information seeking, (FIE’15, EDM’16, ICALT’16)

4. Formative Feedback
   – Augmented automated assessments (ECTEL’16)
Programming learning literature

• **Technology:**
  – ITS (Barnes et al, 2008; Boyer et al, 2011);
  – Adaptive technology (Brusilovsky et al., 2009; Hsiao et al, 2010; Yudelson et al, 2005; Sosnovsky & Brusilovsky, 2015);
  – Social computing (Denny et al, 2011; Hsiao & Brusilovsky, 2011);
  – Automated services (SIGCSE) ...

• **Pedagogy:**
  – Peer instruction (Porter et al., 2013);
  – Media computation (Simon et al, 2010; Guzdial et al, 2013);
  – Flipped classroom (Sarawagi, 2013);
  – Classroom orchestration (Martinez-Maldonado et al, 2013; Slotta, et al, 2013) ...

• **Community:**
  – software engineering (Ko. et al., 2006; Stylos & Myers, 2006; Brandt, 2010);
  – online communities (Glassman et al., 2015; Rivers & Koedinger, 2015; Hsiao & Naveed, 2015)...

**Learning to program is NOT easy!**
Questions that we’re trying to answer:

• How much do novices learn programming in *Just-In-Time* learning paradigm?

• Will integrate *formal* and *informal* learning analytics bridge the gap between cyber and physical programming classes?

• How to design *intelligent* automated assessment tools to assist computing education? (i.e. examples in math)
The reality is...

• Majority of lower division CS courses are still Blended!
  – face-to-face instruction in the classrooms supported by online tools

• Weak link between assessment and instruction!
1. Semantic topic modeling + visualization
Theoretical Background

- Accountable Talk (Resnick, L.)
- ICAP (Interactive, Constructive, Active, Passive) learning activity framework (Chi, M.T.H.)

Vialogues in EdLab@Columbia Univ.

- Computational Modeling
- Social Communicative Factor
- Visualization
- Informal Learning Activity
How to facilitate novices learn programming *informally*?
Joel Spolsky: “Just create the environment, people will come to the environment and behave according to what you build.”
StackOverflow 3 years data: Topic: Java
What are the students reading? How do those content contribute learning?

- Syntactic
- Semantics
- Pragmatics

(b) Programming Knowledge Mapping

Posts’ Semantics

(a) Semantic Content Modeling

(c) Behavior Modeling

Procedural / Declarative Knowledge

{ Reading, Searching, Locating, Evaluating, … }
a. Label “constructive” content (CSCL’15, ICALT’15, FIE’15)
   - inferring, creating, integrating, elaborating, comparing, contrasting, analogizing, generalizing, including, reflecting on conditions, explaining why something works...
   - MPQA (Multiple Perspective Question Answering) opinion corpus developed by U Pitt
     - Emphasis: “…this is why…”
     - Causation: “…as a result…”
     - Generalization: “…everything…”
     - Conditionals: “…it would be…”
   - Positive correlation between user favorites and the constructiveness in Accepted Answers ($r=0.0781$, $p<0.01$)
   - User bookmarked and up voted more in Answers when the content is more constructive.

b. Label content associated concepts

c. Collect students’ learning activities
a. Label “constructive” content

b. Label content associated concepts (LAK’15,16, Computers in Human Interaction: Technology Behaviors in Education Innovation, under review)
   – Semi-auto indexing interfaces: Natural + Programming language parser (topic facet modeling)
   – 2 teachers & MTurk
     • 103/149/406 MTurkers passed & completed the qualifier survey ($0.5 dollars for 2.5 weeks)
       – Each of question in the corpus is indexed by at least 2 to 5 experts from the crowd
       – Can extract sig. more and diverse concepts.

c. Collect students’ learning activities
1. Label “constructive” content
2. Label content associated concepts
3. Collect students’ learning activities (EDM’16)
   – Novices are not developers!
   – The more they read, the more they learned ($r=0.418, p<0.01$)
2. Strengthen the link between Assessment and Instruction
Make students improve from feedback and keep working!

• The Power of Feedback: feed up, feed back, and feed forward (Hattie & Timperley, 2007)
  – To reduce discrepancies between current understandings and performance and a goal.

• Success from ITS:
  – Inner loop feedback: next step
  – Outer loop feedback: problem selection (VanLehn, 2006)
Today’s blended programming classes:

• Formal:
  – Automated submission assignments
  – Paper-based exams

• Informal
  – Optional readings, tools, exercises...
Problems & Goals:

• Problems:
  – *how do students receive partial credits?*
  – *was it a single concept or multiple concepts mistake?*
  – *a careless mistake or a long-term misconception?*
  – *Limited class time (no personalized feedback)*
  – *Graders’ training or inconsistency issue*
  – …

• Goals: using **visual learning analytics** to capture & reinforce
  – identification of strength and weakness
  – characterization of the nature of their errors or any recurring patterns if any
  – assessment of appropriateness of their study strategies and preparation
3 studies & counting...

• Random sample 20 exams 40 questions graded by 6 graders

• User study
  – 14 graders
  – 30 exams

• Classroom study w/ Augmented Grading Tools
Graders can be VERY inconsistent!

Average Total + and - Points Off Per Question

<table>
<thead>
<tr>
<th>Grader</th>
<th>+ Points</th>
<th>- Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>grader1</td>
<td>39.5</td>
<td>9</td>
</tr>
<tr>
<td>grader2</td>
<td>43</td>
<td>6.5</td>
</tr>
<tr>
<td>grader3</td>
<td>32.5</td>
<td>21</td>
</tr>
<tr>
<td>grader4</td>
<td>30.5</td>
<td>5</td>
</tr>
<tr>
<td>grader5</td>
<td>36</td>
<td>5.5</td>
</tr>
<tr>
<td>grader6</td>
<td>23</td>
<td>6.5</td>
</tr>
</tbody>
</table>
1. Write the following for loop as a while loop: (5pt)

```java
for (int i = 0; i <= 10; i++)
{
    System.out.println(i);
}
```

Correct Answer:
```
int i = 0;
while (i <= 10){
    System.out.println(i);
    i++;
}
```

Errors:
- -1 incorrect initialization
- -3 incorrect while statement
- -1 incorrect increment statement
- -0.5 other errors

2. Instantiate an ArrayList that contains decimal numbers and assign it to an appropriate variable. Write an enhanced for loop (for-each loop) that iterates through your ArrayList of decimal numbers and displays their sum to the console: (7pt)

```java
// -2 incorrect ArrayList
ArrayList<Double> numList = new ArrayList<Double>();
double sum = 0;

// -4 incorrect enhanced for-loop statement
for (Double d : numList){
    sum += d;
}
System.out.println(sum);
```

Errors:
- -2 incorrect ArrayList
- -4 incorrect enhanced for-loop statement
- -0.5 other errors

15. Instantiate an ArrayList that contains decimal numbers and assign it to an appropriate variable. Write an enhanced for loop (for-each loop) that iterates through your ArrayList of decimal numbers and displays their sum to the console: (7pt)

```java
double sum = 0;
for (Double element : decimalNumbers)
{
    sum += element;
}
System.out.println(sum);
```
Feedback Type Distribution

- **Cue**: 56.32% (Easy), 52.93% (Complex)
- **Diagnostic**: 27.92% (Easy), 15.19% (Complex)
- **Corrective**: 11.06% (Easy), 9.06% (Complex)
- **Reinforcement**: 2.27% (Easy), 3.70% (Complex)
- **None**: 18.17% (Easy), 3.01% (Complex)
- **Motivational**: 0.38% (Easy), 0.00% (Complex)
- **Negative**: 0.00% (Easy), 0.00% (Complex)

SIGCSE’17 stay tuned!
Approaches to address above issues:

• Consistent *feedback* to students
• Grader support
Augmented Grading Tools: PGA & WPGA

• (Web-based) Programming Grading Assistant
2. Instantiate an ArrayList that contains decimal numbers and a variable. Write an enhanced for loop (for each loop) that iterates over the ArrayList and displays the sum to the console. (7 points)

```java
ArrayList num = new ArrayList();
for (double d : num) {
    // Code here
}
```
**EduAnalysis**: Visual Learning Analytics for both teachers & students

- Supports **indexing, authoring, delivery**
• Persistent traces of learning analytics in semantic level.

• Conceptual feedback for students.

• Semantic Partial Credit Assignment
  – “award the logic soundness instead of code completeness”
Summary & Future On-going Work

• Establish infrastructure to collect formal/informal programming assessments
  • Semantic indexing methods (to label programming-related content to programming topic)
  • Practical solutions (minimum intrusion for class orchestration) to facilitate (large-scaled) blended programming classes

• Conduct more exhaustive studies to examine effects on learning
  • How to provide creative way to excite students in using feedback?
  • How do students prepare exams? cheat sheets? How can I help?
  • How does distributed (bite-size) practice strategy help in programming learning?
#CS4All is real.
It is happening.
We take it very seriously.
We do not take today's programming classes for granted.

Thank you, Questions?
<table>
<thead>
<tr>
<th>Question</th>
<th>Date of Quiz</th>
<th>Your Answer</th>
<th>Check Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrement operator, --, decreases value of variable by what number</td>
<td>2016-06-15</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>How many ports of TCP/IP are reserved for specific protocols?</td>
<td>2016-06-26</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>What is a class?</td>
<td>2016-06-17</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>What is the output of relational operators?</td>
<td>2016-06-16</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>What is the range of data type short in Java</td>
<td>2016-06-13</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>What is the return type of Constructors?</td>
<td>2016-06-20</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>What kind of language is java</td>
<td>2016-06-12</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>What kind of variables a class can consist of?</td>
<td>2016-06-18</td>
<td>Correct</td>
<td></td>
</tr>
</tbody>
</table>
A Check Question

Q) What is the output of relational operators??

A) Integers
B) Boolean
C) character
D) Double

Your selected choice is (C) character is INCORRECT.

The correct choice is (B) Boolean.

No recommended readings to show

---

Now that you understand variables and operators, it’s time to learn about expressions, statements, and blocks. Operators may be used in building expressions, which compute values: expressions are the core components of statements; statements may be grouped into blocks. An expression is a construct made up of variables, operators, and method invocations, which are constructed according to the syntax of the language, that evaluates to a single value. You’ve already seen examples of expressions, illustrated in bold below: the data type of the value returned by an expression depends on the elements used in the expression. The expression `e = 0` returns an int because the assignment operator returns a value of the same data type as its left-hand operand. In this case, `a` is an int. As you can see from the other expressions, an expression can return other types of values as well, such as boolean or string. The Java programming language allows you to construct compound expressions from various smaller expressions as long as the data type required by one part of the expression matches the data type of the other. Here’s an example of a compound expression: In this particular example, the order in which the expression is evaluated is unimportant because the result of multiplication is independent of order; the outcome is always the same, no matter in which order you apply the multiplications. However, this is not true of all expressions. For example, the following expression gives different results, depending on whether you perform the addition or the division operation first: You can specify exactly how an expression will be evaluated using balanced parentheses, e.g., `1 + 2 * 3`. If you have the previous expression ambiguous, you could write the following: If you don’t explicitly indicate the order for the operations to be performed, the order is determined by the precedence assigned to the operators. Statements also have precedence. Therefore, the following two statements are equivalent: When writing compound expressions, be explicit and indicate with parentheses which operators should be evaluated first. This practice makes code easier to read and to maintain. Statements are roughly equivalent to sentences in natural languages. A statement forms a complete unit of execution. The following types of expressions can be made into a statement by terminating the expression with a semicolon (;). Such statements are called expression statements. Here are some examples of expression statements. In addition to expression statements, there are two other kinds of statements: declaration statements and control flow statements. A declaration statement declares a variable: you’ve seen many examples of declaration statements already. Lastly, control