A Teaching System To Learn Programming:
The Programmer’s Learning Machine

Martin Quinson, Gérald Oster
Lorraine University, France

July 8., 2015
itiCSE 2015, Vilnius
The Programmer’s Learning Machine

- Interactive Exerciser to Learn Programming
- Loosely tutored practical sessions at University, College

Short Feedback Loop

http://www.loria.fr/~quinson/PLM
The Programmer’s Learning Machine

- Interactive Exerciser to Learn Programming
- Loosely tutored practical sessions at University, College

Read the mission. Inspect the initial state

http://www.loria.fr/~quinson/PLM
The Programmer’s Learning Machine

- Interactive Exerciser to Learn Programming
- Loosely tutored practical sessions at University, College

Compare to the objective state. Check the demo?

http://www.loria.fr/~quinson/PLM
The Programmer’s Learning Machine

- Interactive Exerciser to Learn Programming
- Loosely tutored practical sessions at University, College

Write your code; Run it: it moves!

http://www.loria.fr/~quinson/PLM
Yet Another IDE for Learners??

- **Exerciser ≠ IDE**: Many exercises, Coherent progressions
- **Multi-language**: Java/Scala/Python (+ C/javascript/Blockly/Ruby)
- **Multi-lingual**: English/French/Brasilian (+ Italian/Chinese)
- **Multi-universe**: Several kind of Micro-Worlds

http://www.loria.fr/~quinson/PLM
Yet Another IDE for Learners??

- Exerciser ≠ IDE: Many exercises, Coherent progressions
- Multi-language: Java/Scala/Python (+ C/javascript/Blockly/Ruby)
- Multi-lingual: English/French/Brasilian (+ Italian/Chinese)
- Multi-universe: Several kind of Micro-Worlds

http://www.loria.fr/~quinson/PLM
Teaching Material: CS0

Tactical Programming

▶ Solve all initial syntax difficulties
▶ Don’t get distracted when the algorithms become hairy

Imperative Kernel (regardless of the language)

▶ Instructions & Comments; Conditionals
▶ while loops; switch cases
▶ Variables
▶ for loops and do/while
▶ Methods, Functions, Parameters
▶ Arrays
▶ Many application and Summative exercises

200 scenarized exercises (10 to 50+ hours)

http://www.loria.fr/~quinson/PLM
Teaching Recursion in CS1

Recursion made Simple

- Split problem; Have a friend solve sub-problem; Gather solutions
Teaching Recursion in CS1

Recursion made Simple

- Split problem; Have a friend solve sub-problem; Gather solutions

Supporting Micro-worlds

- Recursive Lists: `length()`, `isMember()`, etc
- Turtle Recursion: Trees, Fractals
  - `subtree()` provided, toward decomposition
- Hanoi and variations toward decomposition
  - Linear, Cyclic, Bicolor, Tricolor, etc

Initial Settings  | Intermediate Step  | Final Goal
Teaching Sorting in CS1

Discover: go through a dozen algorithms (enforces #operations)

- State view, and History view to help understanding

Apply: Dutch Flag, Pancakes or Pebble Motion Problem
Students’ Feedback

- The PLM is educational. 6% (92%) 2%
- Using the PLM is entertaining. 8% (89%) 4%
- I would advise the PLM to a friend. 8% (87%) 6%
- A practical is more motivating when it leverages the PLM. 15% (83%) 2%
- I progressed thanks to the PLM. 26% (70%) 4%
- The PLM is a good argument to come to our school. 30% (45%) 25%
Creating new Material

- Lesson: Just a graph of exercises (list + dependencies)

```java
public class SlugTrackingEntity
    extends SimpleBuggle {

    // Some additional (hidden) code
    /* BEGIN TEMPLATE */
    boolean isFacingTrail() {
        // Write your code here
        /* BEGIN SOLUTION */
        if (isFacingWall())
            return false;
        forward();
        boolean res =
            getGroundColor().equals(Color.green);
        backward();
        return res;
        /* END SOLUTION */
    }
    /* END TEMPLATE */
}
```

Micro-world: World + View + Entity + ControlPanel

400 to 1400 lines

http://www.loria.fr/~quinson/PLM
Creating new Material

Lesson: Just a graph of exercises (list + dependencies)

Exercise: Mission + Initial Worlds + Correcting Entities

```java
public class SlugTrackingEntity
    extends SimpleBuggle {

    // Some additional (hidden) code
    boolean isFacingTrail() {
        throw new NotImplementedException();
    }

    // Write your code here
    boolean isFacingWall() {
        throw new NotImplementedException();
    }

    if (isFacingWall())
        return false;

    forward();
    boolean res =
        getGroundColor().equals(Color.green);
    backward();
    return res;

    // END SOLUTION */
    // END TEMPLATE */
```
Creating new Material

Lesson: Just a graph of exercises (list + dependencies)

Exercise: Mission + Initial Worlds + Correcting Entities

public class SlugTrackingEntity
    extends SimpleBuggle {

    // Some additional (hidden) code
    /* BEGIN TEMPLATE */
    boolean isFacingTrail() {
        /* BEGIN SOLUTION */
        if (isFacingWall())
            return false;
        forward();
        boolean res = getGroundColor().equals(Color.green);
        backward();
        return res;
    }
    /* END SOLUTION */
    /* END TEMPLATE */

    /* END TEMPLATE */

Micro-world: World + View + Entity + ControlPanel

400 to 1400 lines

http://www.loria.fr/~quinson/PLM

The Programmer's Learning Machine (itiCSE'15, Vilnius)
Helping Researchers on CSER

Do you want to share your data?

The PLM currently saves your usage data anonymously on its servers. That way, you can retrieve your session from any connected computer as long as you sign in.

This anonymous data can also help scientists to understand how people learn programming. With that in mind, we would like to provide your usage data and make it public. As you can see here, no nominative information is published, only the source code written to solve the challenges.

Do you accept to make public your usage data?

- Let me decide later
- No, don’t publish my data
- Ok, share my data

You can later update your preference from your profile’s page.

► Each attempt is saved anonymously, so that it can be rerun
► 133,636 code submissions (as of yesterday):
  ► 17,480 success; 42,693 compilation errors; 73,463 failures
  ► Scala: 87,514 (66%); Python: 28,066 (21%); Java: 16,669 (13%)

http://www.loria.fr/~quinson/PLM
Exhibiting Longer Learning Path

Cumulative count of students that passed more than X exercises

- 863 students passed at least one exercise (all languages)
- 499 students attempted more than 5 exercises
- 436 students passed more than 5 exercises
- #(attempt>25) = 138
- #(pass>25) = 116
- #(pass>50) = 72
- #(pass>100) = 25
- #(pass>150) = 5

(as of July 7, 2015)
Exhibiting Longer Learning Path

Cumulative count of students that wrote more than X lines of code

863 students wrote at least one line
688 students wrote more than 10 lines
452 students wrote more than 50 lines
325 students wrote more than 100 lines
98 students wrote more than 500 lines
56 students
13 students
2 students

(as of July 7, 2015)

▶ https://github.com/BuggleInc/PLM-data
▶ We provide a crawler — contact me.
Current Work on the PLM

From a Java application to a Web-based service
- Tests, Distributed Judges and Dockers ongoing
- Currently at watershed! In production in September

Teacher Console
- Dashboard of Class Progress
- Alerts: Detect stuck students (hard!)

Personalized Remediation for the MOOCs
- Clusterize the logic errors (not compile errors)
- Provide an adapted text to common ones, shown automatically
Conclusions

The PLM helps several audiences

- **Learners:** work at their own peace with a serious «game»
- **Teachers:** automated+monitoring → more time with students
- **Authors:** reuse of non-functional code, instrumented feedback
- **Researchers:** corpus of data

Future Works:

- New worlds, new exercises, new languages, polishing
- Integrated Exercise and Material Editor
- Integrated Q&A System, community-based programming resource

Join us!

Martin.Quinson@loria.fr