Qualitative analysis of Mastery Checks in a programming course

Relatore USI: Prof. Matthias Hauswirth
Relatore UniMiB: Prof. Giovanni Denaro

Laureando: Luca Chiodini
Write a program that will read in integers and output their average. Stop reading when the value 99999 is input.
Students’ performance on Rainfall problem after CS1

- Soloway et al., 1980s: 17% correct, 83% wrong
- McCracken et al., 2000s: 21% correct, 79% wrong
Students’ performance on Rainfall problem after CS1

Soloway et al. 1980s

17% correct
83% wrong

McCracken et al. 2000s

21% correct
79% wrong

Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course
Students’ performance on Rainfall problem after CS1

Soloway et al. 1980s
17% correct
83% wrong

McCracken et al. 2000s
21% correct
79% wrong
Computer Science Education

• Interdisciplinary research area
  • Pedagogy, learning and cognitive sciences
  • Core CS fields (e.g., programming languages)
Interdisciplinary research area
  - Pedagogy, learning and cognitive sciences
  - Core CS fields (e.g., programming languages)

Find better *means* and *strategies* to enable students to master CS topics
Computer Science Education

- Interdisciplinary research area
  - Pedagogy, learning and cognitive sciences
  - Core CS fields (e.g., programming languages)
- Find better *means* and *strategies* to enable students to master CS topics
- ACM SIGCSE conferences (ITiCSE, ICER)
We want to understand:

- which misconceptions students develop
- which strategies are used to tackle problems
- how learning trajectories evolve over time
Qualitative study

We want to understand:

- which misconceptions students develop
- which strategies are used to tackle problems
- how learning trajectories evolve over time

Study outline:

- recruited 6 first-year students attending Programming Fundamentals 2
- held and recorded 10 individual Mastery Check sessions (roughly 30’)
- collected over 1600 minutes covering a wide range of Java topics
A new tool to sync everything automatically...

Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course
Huge amount of manual tedious work
Find audio offsets to align recordings (using cross-correlation)
Pipeline

Auto-detect segments with motion analysing frames’ differences (using OpenCV)
Embed transcribed text as subtitles
(forced alignment)
Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course

```java
public class Calculator {
  private double value;

  public Calculator(){
    this.value = 0;
  }

  public void add(double x){
    this.value = x + value;
  }

  public void clear(){
    this.value = 0;
  }

  public double get(){
    return this.value;
  }

  public void set(double y){
    value = y;
  }

  Yeah, if I have value here, I could use it. I have to use this.
}
```
Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course
200+ codes divided into 9 macro categories
200+ codes divided into 9 macro categories

- **MISCONCEPTIONS**
  Specific wrong conceptions about syntax or semantics of Java.
200+ codes divided into 9 macro categories

- **MISCONCEPTIONS**
  Specific wrong conceptions about syntax or semantics of Java.

- **STRATEGYERRORS**
  Bad processes used by novices to solve tasks.
200+ codes divided into 9 macro categories

- **MISCONCEPTIONS**
  Specific wrong conceptions about syntax or semantics of Java.

- **STRATEGYERRORS**
  Bad processes used by novices to solve tasks.

- **METAINFOABOUTERRORS**
  Additional information about errors (inconsistencies, patterns, novelty, ...)

Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course
200+ codes divided into 9 macro categories

- **MISCONCEPTIONS**
  Specific wrong conceptions about syntax or semantics of Java.

- **STRATEGYERRORS**
  Bad processes used by novices to solve tasks.

- **METAINFOABOUTERRORS**
  Additional information about errors (inconsistencies, patterns, novelty, ...)
### Description of a misconception

**Title** SuperclassObjectIsAllocated  
**Context** Class Child extends class Parent  
**Description** When new Child() is executed, two objects are created: a Child object with the fields that belong to the class Child and a Parent object with the fields that belong to the class Parent.  
**JLS** §8.2 Class Members  
**Observations** Sessions 8 and 10
Example of a misconception

**Example Code**

```java
public class Empolyee {
    private int dailySalary;
    ...
}

public class ProjectManager extends Empolyee {
    private int bonus;
    ...
}
```

Luca Chiodini, Qualitative analysis of Mastery Checks in a programming course
Insights from coded segments

• Some misconceptions are one the dual of another
  • Duality in the type system: IMPLICITNARROWING vs NOIPLICITWIDENING
  • Duality in “collection” types: ARRAYHASLENGTHMETHOD vs STRINGLENGTHFIELD
Some misconceptions are one the dual of another

- Duality in the type system: `IMPLICITNARROWING` vs `NOIMPLICITWIDENING`
- Duality in “collection” types: `ARRAYHASLENGTHMETHOD` vs `STRINGLENGTHFIELD`

Some misconceptions are caused by wrong analogies

- `ARRAYLISTELEMENTACCESSUSINGQUAREBRACKETS`
- `ARRAYSHAVECONSTRUCTOR`

Tackling a problem the right way is hard

- ThinkingAlgoComplexityBeforeSimpleAndCorrect
- MissingBaseCaseInRecursion
- MissingReturnInRecursion
- NotRelyingOnInductionInRecursion
Insights from coded segments

- Some misconceptions are one the dual of another
  - Duality in the type system: *implicitNarrowing* vs *NoImplicitWidening*
  - Duality in “collection” types: *arrayHasLengthMethod* vs *stringLengthField*
- Some misconceptions are caused by wrong analogies
  - *ArrayListElementAccessUsingSquareBrackets*
  - *ArraysHaveConstructor*
- Tackling a problem the right way is hard
  - *thinkingAlgoComplexityBeforeSimpleAndCorrect*
  - *missingBaseCaseInRecursion*
  - *missingReturnInRecursion*
  - *notRelyingOnInductionInRecursion*
Insights from coded segments on Notional Machines

**Notional Machine (Fincher et. al., 2020)**
A notional machine is a pedagogic device to assist the understanding of some aspect of programs or programming.
Insights from coded segments on Notional Machines

Notional Machine (Fincher et. al., 2020)
A notional machine is a pedagogic device to assist the understanding of some aspect of programs or programming.

- NOTIONAL_MACHINE HELPS RECOGNIZING ERROR
- NOT DOING STEPS IN ORDER IN STACK AND HEAP DIAGRAM
Notional Machine (Fincher et. al., 2020)
A notional machine is a pedagogic device to assist the understanding of some aspect of programs or programming.

- NOTIONAL_MACHINE HELPS RECOGNIZING ERROR
- NOT DOING STEPS IN ORDER IN STACK AND HEAP DIAGRAM

`IntHolder h1 = new IntHolder(5);`
## Learning trajectories

<table>
<thead>
<tr>
<th>Session</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 6</td>
<td>Correct</td>
<td>Wrong</td>
<td>N/A</td>
<td>Wrong</td>
<td>N/A</td>
</tr>
<tr>
<td>Session 8</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
</tr>
<tr>
<td>Session 9</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Correct</td>
<td>Wrong</td>
</tr>
<tr>
<td>Session 10</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
</tr>
</tbody>
</table>

**Table 1:** Correctness of `THISEXISTSINSTATICMETHOD` across four sessions.
# Learning trajectories

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 6</td>
<td>Correct</td>
<td>Wrong</td>
<td>N/A</td>
<td>Wrong</td>
<td>N/A</td>
</tr>
<tr>
<td>Session 8</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
</tr>
<tr>
<td>Session 9</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Correct</td>
<td>Wrong</td>
</tr>
<tr>
<td>Session 10</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
</tr>
</tbody>
</table>

**Table 1:** Correctness of `THISEXISTSINSTATICMETHOD` across four sessions.

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 8</td>
<td>Correct</td>
<td>Correct</td>
<td>Wrong</td>
<td>Wrong</td>
<td>Wrong</td>
</tr>
<tr>
<td>Session 10</td>
<td>Correct</td>
<td>Correct</td>
<td>Correct</td>
<td>Correct</td>
<td>Wrong</td>
</tr>
</tbody>
</table>

**Table 2:** Correctness of `SUPERCLASSOBJECTISALLOCATED` across two sessions.
Conclusions and follow-up studies

Exploratory phase:

- Developed a useful tool for all qualitative research studies
- Added 100+ newly uncovered Java misconceptions
- Initial attempt to understand learning trajectories
- Observed solving strategies looking at the way students produce artifacts
Conclusions and follow-up studies

Exploratory phase:

• Developed a useful tool for all qualitative research studies
• Added 100+ newly uncovered Java misconceptions
• Initial attempt to understand learning trajectories
• Observed solving strategies looking at the way students produce artifacts

Ideas for possible teaching improvements (subject of future targeted studies):

• Make teachers aware of common misconceptions
• Prepare assessments to detect misconceptions
• Classify misconceptions and strategies across different programming languages
• Know which interventions successfully solve an existing issue