EXAMPLE EXERCISE

Task description:
Implement a class "Queue" that works like a queue (as described in the lecture). The class should have at least the methods isEmpty(), head(), enqueue(x) and dequeue(). The queue does not need to hold more than 100 elements.

Note 1: In the Java test, the stack is expected to store "strings". Since Python is dynamically typed, this does not apply.

Note 2: It is not allowed to use "import"!

A student's Python solution:

```python
class Queue:
    def __init__(self):
        self.array = []
    def isEmpty(self):
        return len(self.array) == 0
    def enqueue(self, x):
        self.array.append(x)
    def dequeue(self):
        return self.array.pop(0)
```

3 out of 10 Python test cases for the task:

<table>
<thead>
<tr>
<th>Test case</th>
<th>Correct output</th>
</tr>
</thead>
<tbody>
<tr>
<td>q = Queue()</td>
<td>True</td>
</tr>
<tr>
<td>print(q.isEmpty())</td>
<td>True</td>
</tr>
<tr>
<td>q = Queue()</td>
<td>False</td>
</tr>
<tr>
<td>for i in range(1,101): q.enqueue(i); q.dequeue()</td>
<td>41</td>
</tr>
<tr>
<td>q.enqueue(&quot;Kakao&quot;)</td>
<td>False</td>
</tr>
<tr>
<td>print(q.isEmpty())</td>
<td>False</td>
</tr>
</tbody>
</table>

CODING & GRADING ENVIRONMENT

Bachelor-level algorithm and data structures courses at Universität Hamburg.

Course work organized in Moodle (https://moodle.org), a free and open-source e-learning platform.


CodeRunner plug-in (coderunner.org.nz) to provide an interactive coding environment within Moodle.

Student can test a potential solution against a subset of test cases defined for each exercise.

Non-visible set of tests featured randomized inputs to prevent hard-coded solutions & checks against forbidden use of library functions.

Consequently, students mostly wrote code from scratch as intended, although copying code from other sources could not be prevented.

DATASET OF STUDENT SOLUTIONS TO ALGORITHM AND DATA STRUCTURE PROGRAMMING ASSIGNMENTS

FYNN PETERSEN-FREY, MARCUS SOLL, LOUIS KOBRAS, MELF JOHANNSEN, PETER KLING, CHRIS BIEMANN

DATASET OVERVIEW

Student solutions to natural language task descriptions → learn to translate natural language to source code.

Enable future research on learning programming, algorithms and data structures both for students and in machine learning contexts.

21 task descriptions in German and English, 533 test cases and 1526 student source code solutions.

Only correct solutions by students who consented on collection and pseudonymised publication.

Statistics for the courses in 2019/20, 2020/21 and 2021/22:

<table>
<thead>
<tr>
<th>Course</th>
<th>19/20</th>
<th>20/21</th>
<th>21/22</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exercises</td>
<td>10</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Students</td>
<td>85</td>
<td>91</td>
<td>128</td>
</tr>
<tr>
<td>Correct solutions (abs.)</td>
<td>541</td>
<td>415</td>
<td>570</td>
</tr>
<tr>
<td>Correct solutions (rel. %)</td>
<td>68.5</td>
<td>75.0</td>
<td>73.3</td>
</tr>
<tr>
<td>Test cases</td>
<td>241</td>
<td>142</td>
<td>150</td>
</tr>
<tr>
<td>Avg. task descr. length</td>
<td>122.7</td>
<td>200.4</td>
<td>201.0</td>
</tr>
<tr>
<td>Avg. LOC</td>
<td>25.3</td>
<td>21.8</td>
<td>16.6</td>
</tr>
<tr>
<td>Avg. LOC (Java)</td>
<td>28.8</td>
<td>26.1</td>
<td>20.0</td>
</tr>
<tr>
<td>Avg. LOC (Python)</td>
<td>19.7</td>
<td>17.7</td>
<td>12.7</td>
</tr>
</tbody>
</table>

DATASET CREATION

Task descriptions, test cases and student solutions are produced in CodeRunner/Moodle making it the single source of truth for the data.

Export all task descriptions, test cases and student solutions in their final state after a course has ended.

Students could freely choose whether they allow the usage and redistribution of their solutions for research purposes.

Student names replaced by random identifier → allow tracing the solutions of a student across exercises.

Original task descriptions in German; we provide English translations for better accessibility.

Task descriptions converted to PDF (via LaTeX) and plain text with LaTeX-math for formulas.

DATASET ANALYSIS

Tasks like exponentiation by squaring, least common multiple, edit distance and data structures such as queue, stack, search trees.

Number of exercises cut in half from the 2019/20 to 2020/21 course; average task description length increased by over 60%.

Percentage of correct solutions increased slightly while average solution length decreased for both Java and Python → newer tasks are not more challenging, but descriptions provide more information.

Percentage of correct solutions varies across tasks:

In the 2020/21 and 2021/22 courses, most exercises state a maximal run time in big-O notation → enforced by a time limit in the CodeRunner plugin.

TAKEAWAY

Dataset of natural language instructions in German and English describing algorithmic programming tasks.

Dozens of correct source code solutions per task.

Many test cases to automatically verify any newly produced solution.

More exercises, solutions and test cases will be added from upcoming courses.

Download: https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/ad-lrec

UESTET
FÜR MATHEMATIK, INFORMATIK UND NATURWISSENSCHAFTEN

Fynns Petersen-Frey:
fyinn.petersen-frey@uni-hamburg.de

DEPARTMENT OF INFORMATICS

LREC 2022
Marseille
June 20–25