Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

FAKULTÄT
FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

DATASET OF STUDENT SOLUTIONS TO ALGORITHM AND DATA STRUCTURE PROGRAMMING ASSIGNMENTS

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

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:

```
class Queue:
    def __init__(self):
        self.array = []
    def isEmpty(self):
        return len(self.array) == 0
    def head(self):
        return 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:

Test case	Correct output
`q = Queue()` `print(q.isEmpty())`	True

`q = Queue()`
`for i in range(1,101):`
`  q.enqueue(i)`
`for i in range(40):`
`  q.dequeue()`
`print(q.head())`
	41

`q = Queue()`
`q.enqueue(”Kakao”)`
`print(q.isEmpty())`
	False

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.

Manual grading of solutions infeasible & need for interactive feedback during coding → **automatic, instantaneous assessment using unit tests**.

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 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:

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

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 DISTRIBUTION

License: **CC BY-NC 4.0** (Creative Commons Attribution-NonCommercial 4.0 International) for **easy sharing and redistribution** of both the original and derived data; only requiring attribution and forbidding commercial usage.

3 CSV files: Task descriptions, test cases and student solutions

Task descriptions: identifier, original German title, translated English title, plain text German task description, translated English task description, solution code skeleton, shared code for the test cases and a solution.

Test cases: Task id, test case number, test code, expected output, example flag.

Student solutions: Task id, randomized student id, solution code.

Task id contains semester, exercise block, exercise number, programming language: e.g. `19_20-4-2-java`

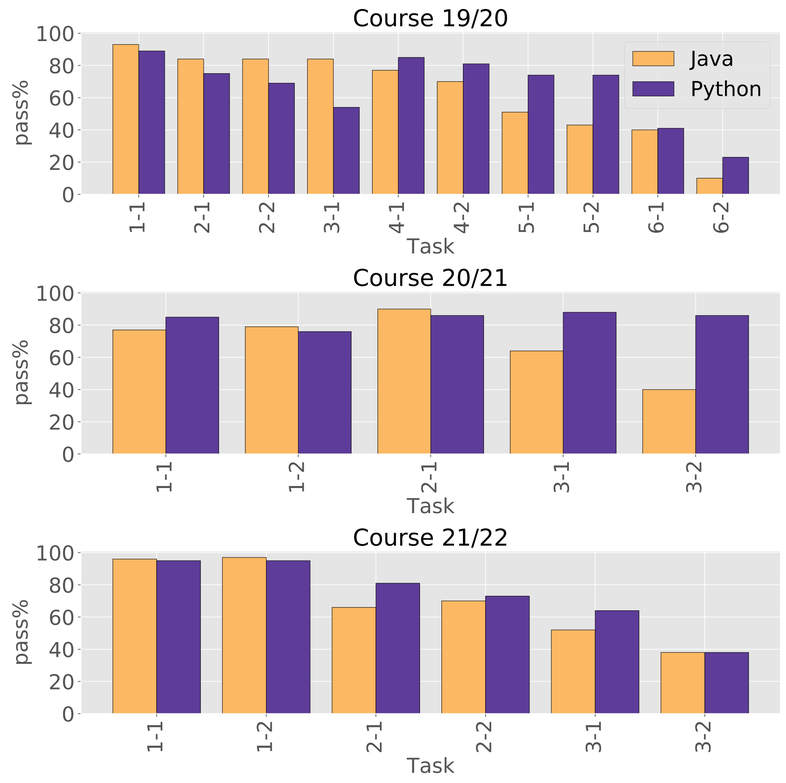
DATASET ANALYSIS

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-$\mathcal{O}$ 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`

DEPARTMENT OF INFORMATICS
Language Technology

LREC 2022
Marseille
June 20–25

Fynn Petersen-Frey
fynn.petersen-frey@uni-hamburg.de