Experience Teaching RISC-V Assembly Programming with Moodle

Idilio Drago[1], **Sergio Rabellino**[1], Enrico Cassano[2], Saverio Accurso[2]
[1] **University of Turin, Computer Science Department**
{idilio.drago,sergio.rabellino}@unito.it
[2] University of Turin, Course degree in Computer Science
{enrico.cassano,saverio.accurso}@edu.unito.it

What the hell is RISC-V ?

RISC-V is an open-source Instruction Set Architecture (ISA) designed to be

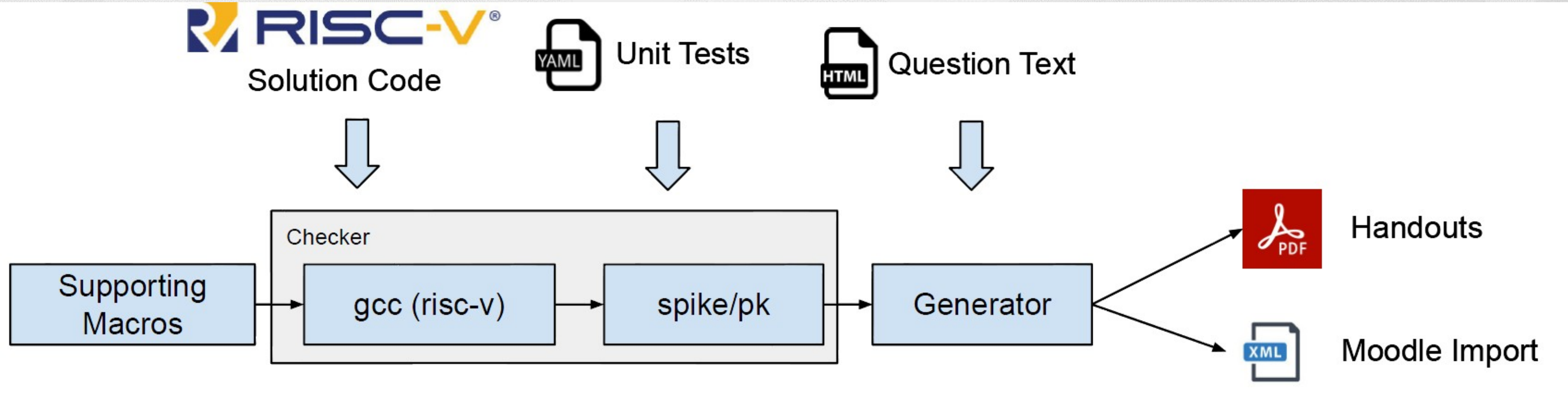
- Simple
- modular
- Extensible
- Versatile choice for a wide range of real applications
- Excellent ISA option for teaching computer architectures and assembly programming

But… Given the middle-level nature of the language, it is challenging for students to test the correctness of their code and run their programs in practice.
This high entry barrier often results in a difficult learning experience, reducing the effectiveness of the learning process.

We will describe our solution for better teaching computer architectures using Moodle:

- We have developed a pipeline to simplify the preparation of programming exercises using the RISC-V language.
- We extended the dockerized version of "Jobe in a Box" to support RISC-V assembly and execution.
- We built a new plugin for Moodle Quiz to enable teachers to download the quiz attempts history autonomously, for analisys purposes

From idea to moodle



Checker: a Python module that takes as input

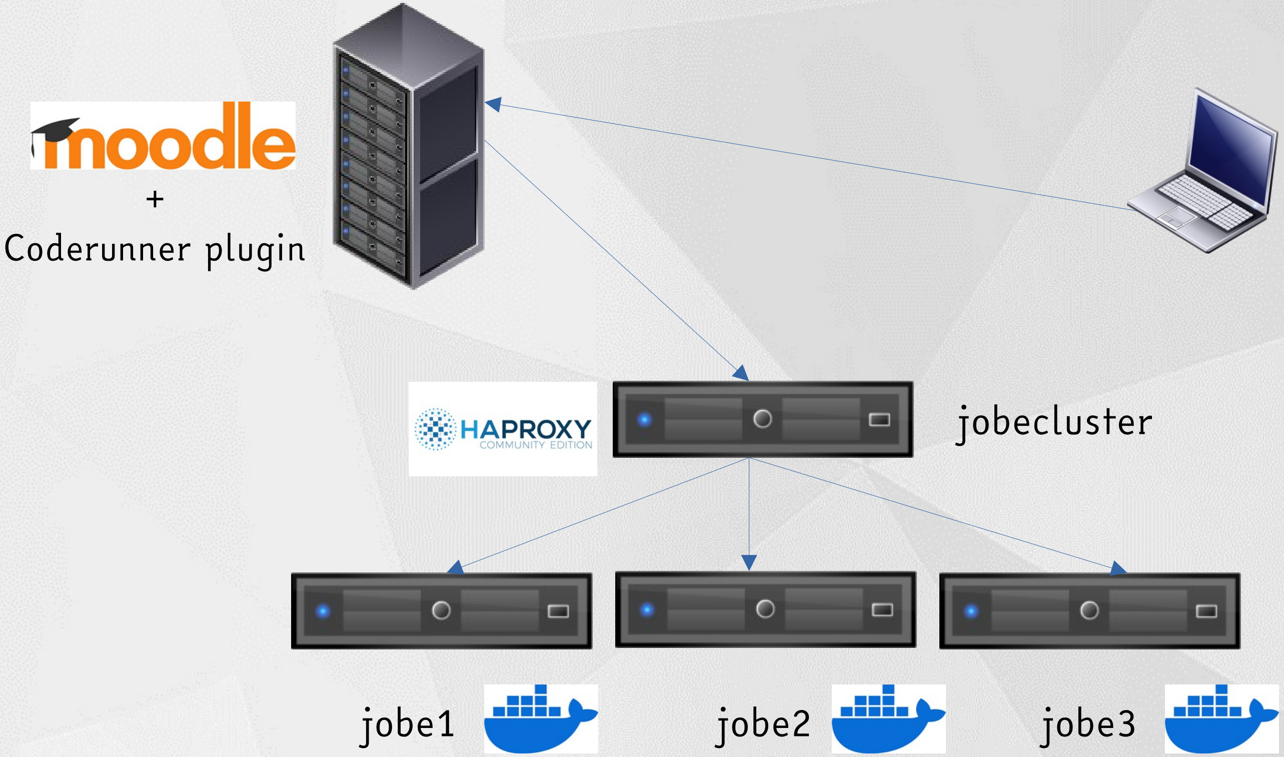
- a RISC-V source code containing the function to be tested and a test driver (main) that calls the particular function,
- a source file with supporting macros (common to all questions), and
- a set of unit tests. It runs all unit tests on the code using the test driver and produces a report of the results in textual form.

Generator: a second Python module that takes as input the same files as above and an additional HTML file containing the question text. It produces PDF handouts (using pandoc) and the XML file to import the question into Moodle.

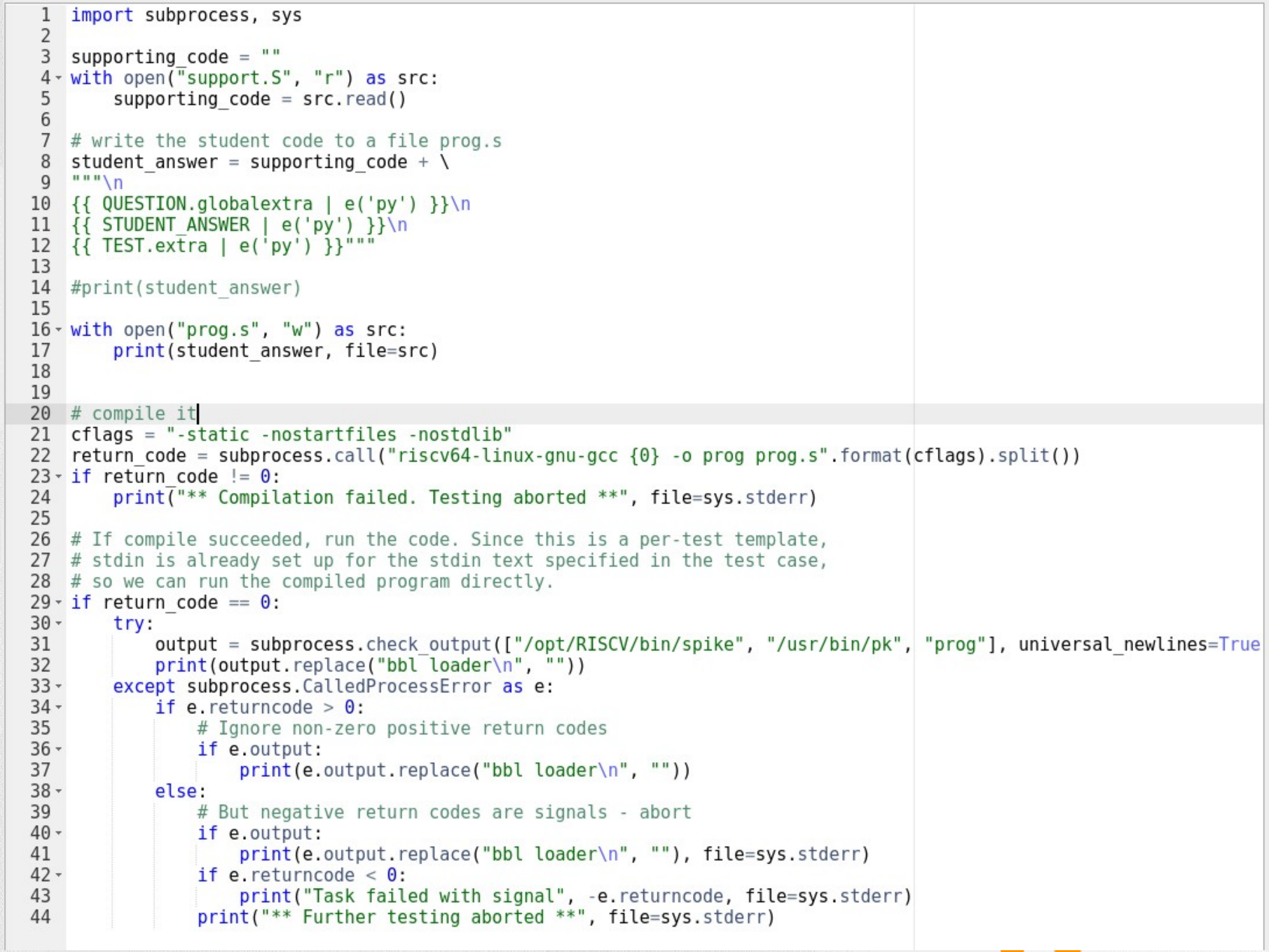
https://github.com/idrago/ArchI-exercise

What changes for RISC-V?

```
#
# For RISC-V purposes
#
RUN apt-get install -y gcc-riscv64-unknown-elf
RUN apt-get install -y gcc-riscv64-linux-gnu
WORKDIR /root
RUN git clone https://github.com/riscv-software-src/riscv-isa-sim
RUN git clone https://github.com/riscv-software-src/riscv-pk
RUN apt-get install -y device-tree-compiler
RUN cd riscv-isa-sim && \
        mkdir build && \
        cd build && \
        ../configure --prefix=/opt/RISCV && \
        make && \
        make install
RUN cd riscv-pk && \
        mkdir build && \
        cd build && \
        ../configure --prefix=/opt/RISCV --host=riscv64-linux-gnu && \
        make && \
        make install
ENV PATH="/opt/RISCV/bin:/opt/RISCV/riscv64-linux-gnu/bin:${PATH}"
RUN cd /usr/bin; ln -s /opt/RISCV/riscv64-linux-gnu/bin/pk .
```



A “riscv” question



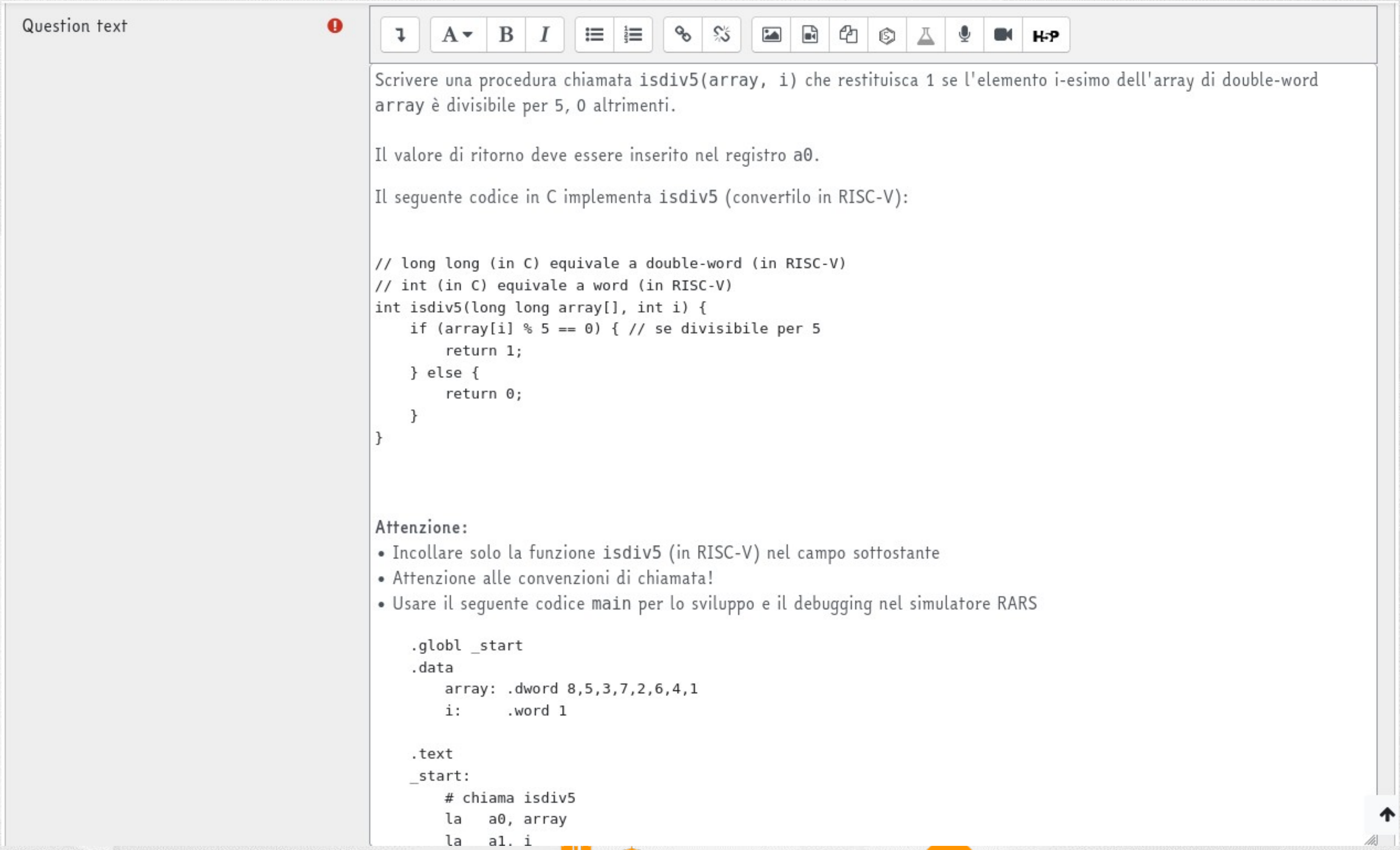
```python
import subprocess, sys

supporting_code = ""
with open("support.S", "r") as src:
    supporting_code = src.read()

# write the student code to a file prog.s
student_answer = supporting_code + \
"""\n
{{ QUESTION.globalextra | e('py') }}\n
{{ STUDENT_ANSWER | e('py') }}\n
{{ TEST.extra | e('py') }}"""

#print(student_answer)

with open("prog.s", "w") as src:
    print(student_answer, file=src)


# compile it
cflags = "-static -nostartfiles -nostdlib"
return_code = subprocess.call("riscv64-linux-gnu-gcc {0} -o prog prog.s".format(cflags).split())
if return_code != 0:
    print("** Compilation failed. Testing aborted **", file=sys.stderr)

# If compile succeeded, run the code. Since this is a per-test template,
# stdin is already set up for the stdin text specified in the test case,
# so we can run the compiled program directly.
if return_code == 0:
    try:
        output = subprocess.check_output(["/opt/RISCV/bin/spike", "/usr/bin/pk", "prog"], universal_newlines=True
        print(output.replace("bbl loader\n", ""))
    except subprocess.CalledProcessError as e:
        if e.returncode > 0:
            # Ignore non-zero positive return codes
            if e.output:
                print(e.output.replace("bbl loader\n", ""))
        else:
            # But negative return codes are signals - abort
            if e.output:
                print(e.output.replace("bbl loader\n", ""), file=sys.stderr)
            if e.returncode < 0:
                print("Task failed with signal", -e.returncode, file=sys.stderr)
            print("** Further testing aborted **", file=sys.stderr)
```

Python takes the RISC-V code, compile it and runs against the test-cases.

If all goes fine, you passed, elsewhere you’ll get some errors...

The Coderunner quest...

A "riscv" question example



Scrivere una procedura chiamata `isdiv5(array, i)` che restituisca 1 se l'elemento i-esimo dell'array di double-word `array` è divisibile per 5, 0 altrimenti.

Il valore di ritorno deve essere inserito nel registro `a0`.

Il seguente codice in C implementa `isdiv5` (convertilo in RISC-V):

```
// long long (in C) equivale a double-word (in RISC-V)
// int (in C) equivale a word (in RISC-V)
int isdiv5(long long array[], int i) {
    if (array[i] % 5 == 0) { // se divisibile per 5
        return 1;
    } else {
        return 0;
    }
}
```

Attenzione:

- Incollare solo la funzione `isdiv5` (in RISC-V) nel campo sottostante
- Attenzione alle convenzioni di chiamata!
- Usare il seguente codice `main` per lo sviluppo e il debugging nel simulatore RARS

```
    .globl _start
    .data
        array: .dword 8,5,3,7,2,6,4,1
        i:     .word 1

    .text
    _start:
        # chiama isdiv5
        la   a0, array
        la   a1, i
```

Students ? Why ?

Question dataset

- Our RISC-V solution have been used in two academic years at the University of Turin to support the course of Computer Architecture in the Computer Science Bachelor's degree.
- In the first year, the system was used primarily for exams, while in the second year, the question dataset was extended and self-assessment quizzes were put online during the semester.

In total, the system has supported more than 1000 students in both exams and self-assessments.

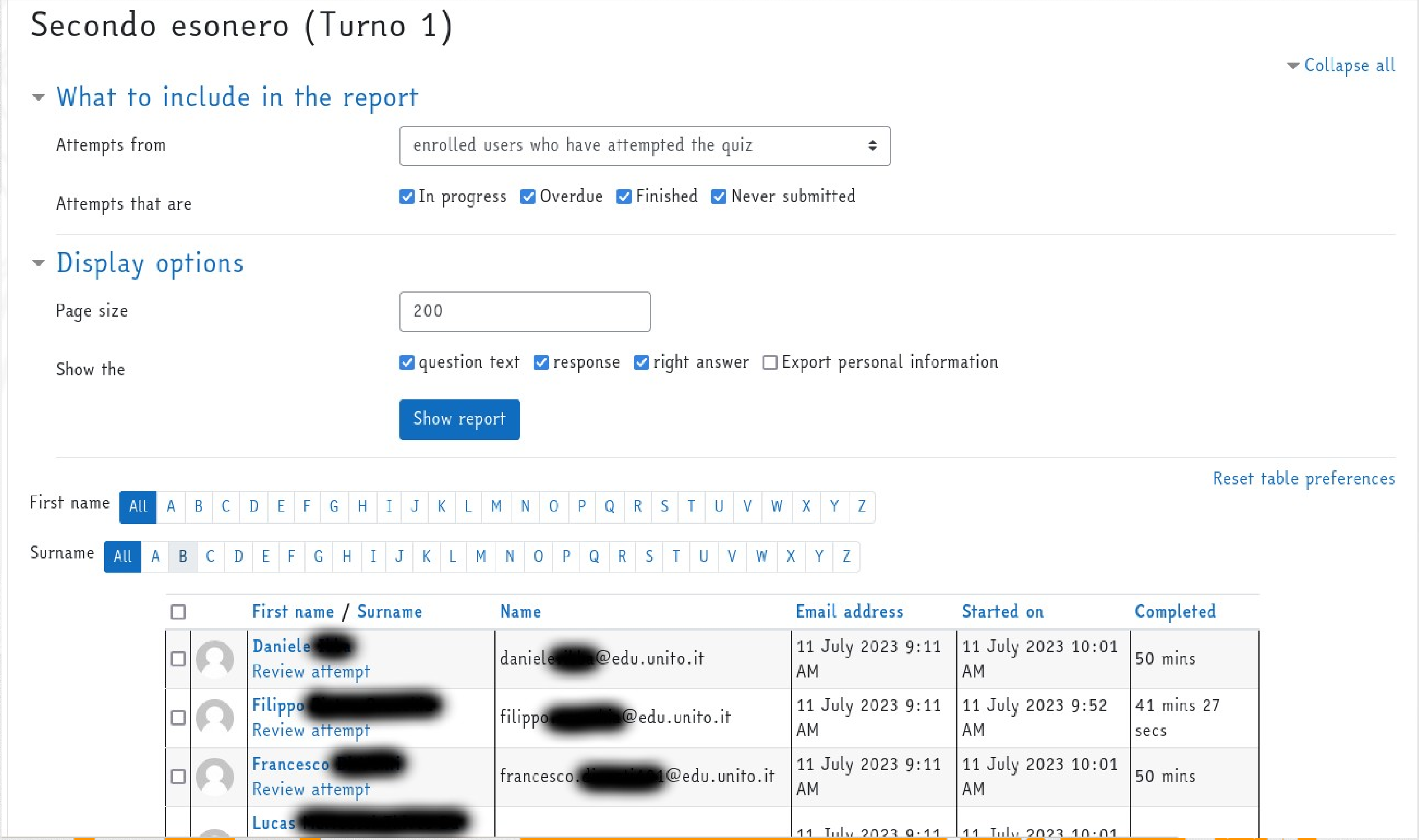
Our programming questions cover a wide spectrum of skills, including arithmetic operations (addition, subtraction, division), logic operations; memory load and store of different data types, control flow instructions (branches and jumps), stack manipulation, bit-wise operations, and addressing (e.g., immediate).

We funded **two students** that cooperated to enhance RISC-V question dataset, starting from the "problem" given by the teachers and developing the solution, the test cases and checked all the things together with Moodle/CodeRunner.

Actually most of the questions are written in Italian, but our goal is to extend and translate all questions to English soon.

Ask Moodle for learning analytics..

The new moodle plugin



Secondo esonero (Turno 1)

▾ What to include in the report

Attempts from: enrolled users who have attempted the quiz

Attempts that are: ☑ In progress ☑ Overdue ☑ Finished ☑ Never submitted

▾ Display options

Page size: 200

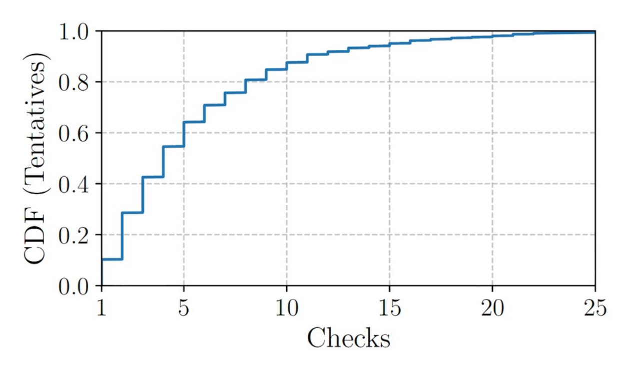
Show the: ☑ question text ☑ response ☑ right answer □ Export personal information

Show report
Reset table preferences
First name All A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Surname All A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

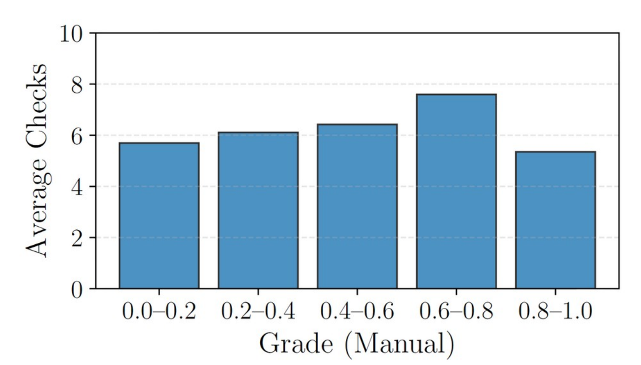
□	First name / Surname	Name	Email address	Started on	Completed
□	Daniele [redacted] Review attempt	daniele[redacted]@edu.unito.it	11 July 2023 9:11 AM	11 July 2023 10:01 AM	50 mins
□	Filippo [redacted] Review attempt	filippo[redacted]@edu.unito.it	11 July 2023 9:11 AM	11 July 2023 9:52 AM	41 mins 27 secs
□	Francesco [redacted] Review attempt	francesco.[redacted]1@edu.unito.it	11 July 2023 9:11 AM	11 July 2023 10:01 AM	50 mins
	Lucas [redacted]		11 July 2023 9:11	11 July 2023 10:01	

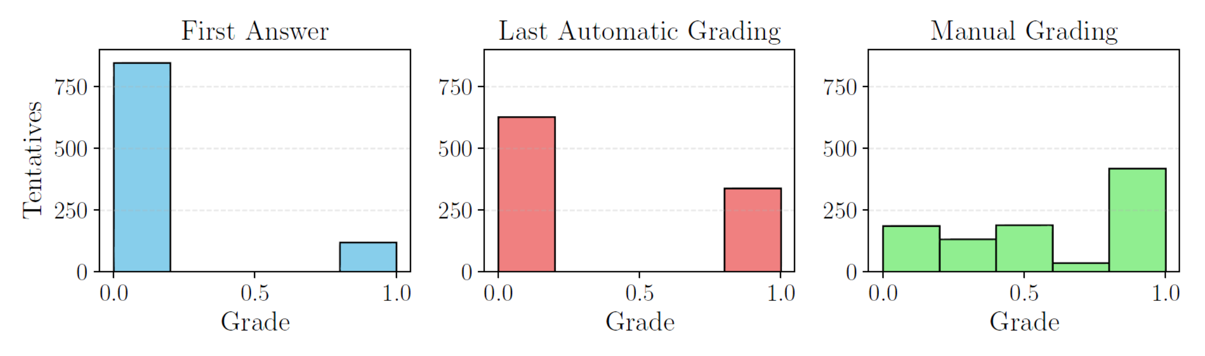
But it's useful ?

What the statistics say



- Analisys from more than 900 students involved in 500 quizzes.
- Students can choose to use an external simulator (RARS), so the Coderunner/Moodle it's an option
- Only GO-NOGO feedback from CodeRunner during exams
- All submission are manually evaluated by teachers (printf("1"))





The end.