# Learning Programming without Teachers: An Ongoing Ethnographic Study at 42


**Nicolò Pollini**
University of Florence

**Kevin Maggi**
University of Florence

**Roberto Verdecchia**
University of Florence

**Enrico Vicario**
University of Florence


UNIVERSITÀ DEGLI STUDI FIRENZE

**DINFO**
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

{ST LAB} Software Technologies Laboratory

# What is 42?

A network for **higher education in Computer Programming**:

- **no tuition fees** & **no school prerequisites**: open to everyone
- **peer-learning**: no teachers, no classes, no courses, no grades
- **video game dynamics**: path based on projects and levels

Duration: 2-3 years (Common Core + Mastery)

Topics: *Bash, Git, procedural programming (C), object-oriented programming (C++), algorithms, data structures, concurrent programming, computer graphics, web and mobile development, …*

# Motivation

SE education nowadays:

- **employment demand**: technology industry continuous expansion
- **ever-evolving environment:** rapid pace of technological advancement
- **traditional education limitations**: costs, classrooms sizes, students needs

# Motivation

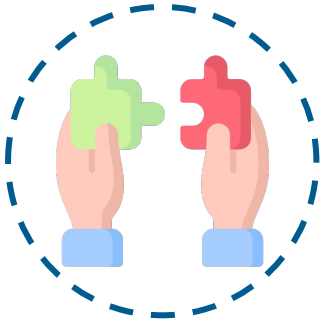
SE education nowadays:

- **employment demand**: technology industry continuous expansion
- **ever-evolving environment:** rapid pace of technological advancement
- **traditional education limitations**: costs, classrooms sizes, students needs

Why 42?

- **established worldwide**: 54 campuses over all the continents
- **atypical educational stance**: no frontal lessons
- **unique mix of modern pedagogical approaches**

# Pedagogy at 42



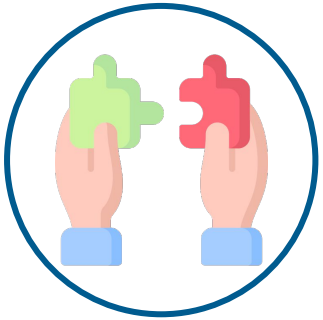
Problem-Based Learning

Gamification

Peer Pedagogy

Community Development

# Problem-Based Learning



Problem-Based Learning

Gamification

Peer Pedagogy

Community Development

# Problem-Based Learning

> *“ Write a library that contains `ft_printf()`, a function that will mimic the original `printf()` ”*

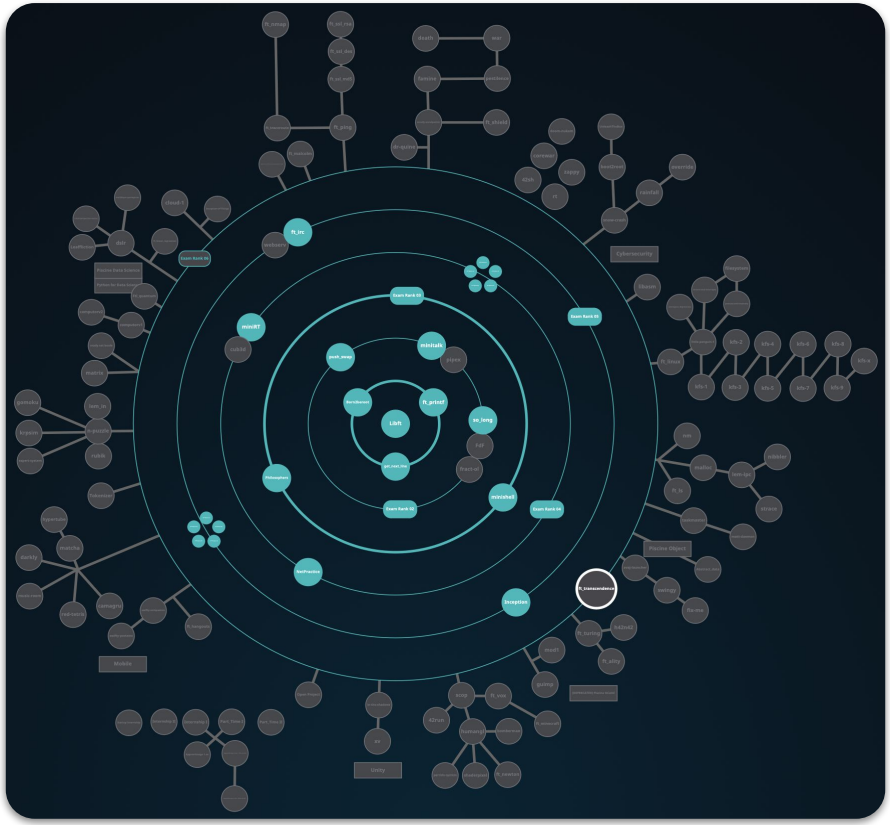
# Problem-Based Learning

> *“ Write a library that contains `ft_printf()`, a function that will mimic the original `printf()` ”*

- **Progression-oriented inductive method**: each topic builds knowledge for future steps

  *e.g. “You must have finished quest(s) common-core-rank-01”*

# Problem-Based Learning



# Problem-Based Learning

> *“ Write a library that contains `ft_printf()`, a function that will mimic the original `printf()` ”*

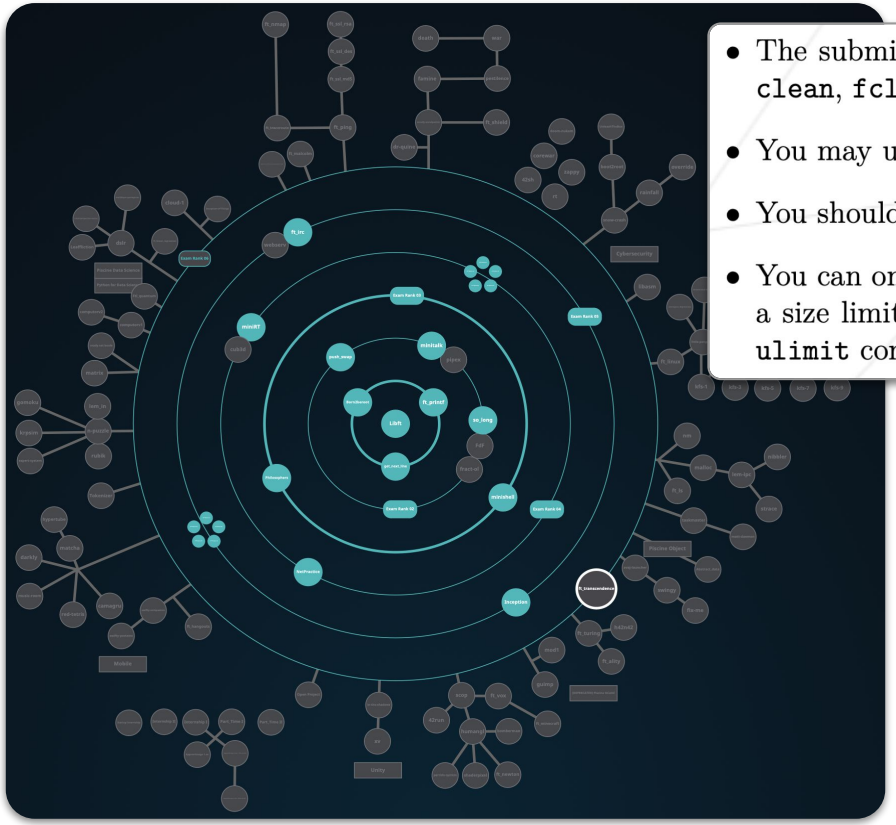
- **Progression-oriented inductive method**: each topic builds knowledge for future steps

  *e.g. “You must have finished quest(s) common-core-rank-01”*

- **Only bare minimum knowledge provided**: student have to learn how to find a solution

  *e.g. “Your reference guide is called Google / man / the Internet / ...”*

# Problem-Based Learning



- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno` (check the `man` for `Errno`).
- You should read the man of all the authorized functions
- You can only do this exercise by declaring a fixed-sized array. This array will have a size limited to a little less than `30 ko`. In order to test that size-limit, use the `ulimit` command-line in your Shell.

# Problem-Based Learning

> *“ Write a library that contains `ft_printf()`, a function that will mimic the original `printf()` ”*

- **Progression-oriented inductive method**: each topic builds knowledge for future steps

  *e.g. “You must have finished quest(s) common-core-rank-01”*

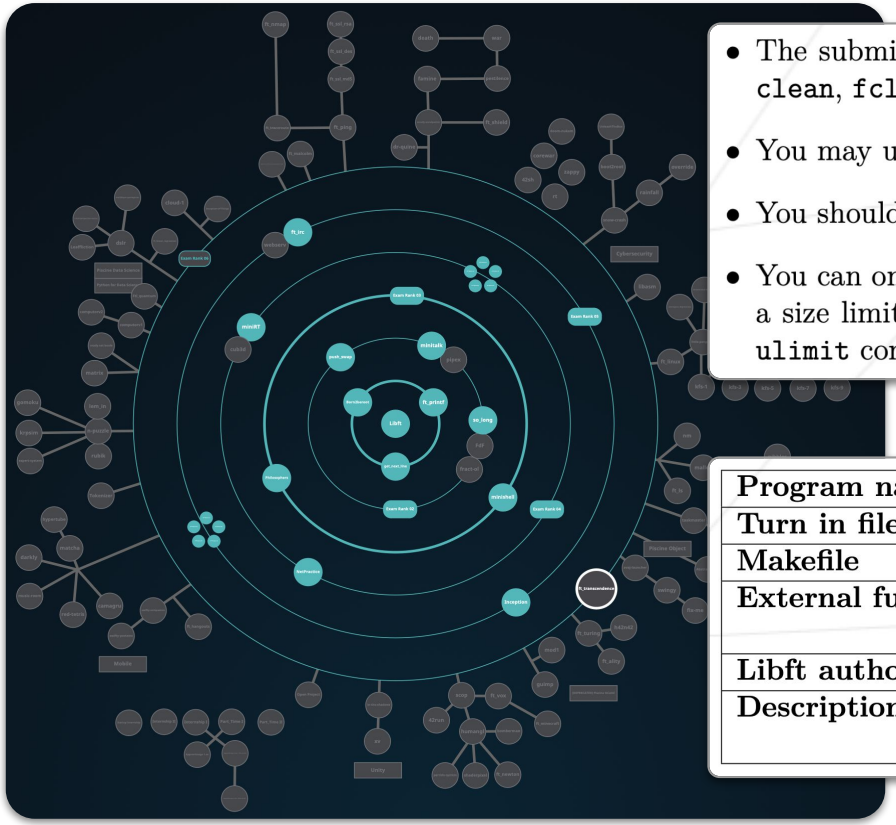
- **Only bare minimum knowledge provided**: student have to learn how to find a solution

  *e.g. “Your reference guide is called Google / man / the Internet / ...”*

- Specific **rules and checks to comply with**

  *e.g. “Your project must comply with the following rules:*
  *- Global variables are forbidden*
  *- In case of error, … ”*

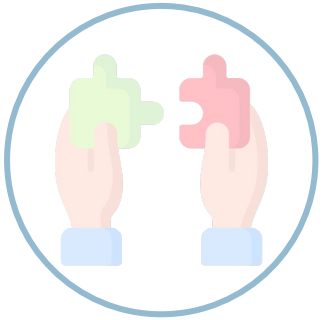
# Problem-Based Learning



- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno` (check the `man` for `Errno`).
- You should read the man of all the authorized functions
- You can only do this exercise by declaring a fixed-sized array. This array will have a size limited to a little less than `30 ko`. In order to test that size-limit, use the `ulimit` command-line in your Shell.

| | |
|---|---|
| **Program name** | `libftprintf.a` |
| **Turn in files** | `Makefile, *.h, */*.h, *.c, */*.c` |
| **Makefile** | `NAME, all, clean, fclean, re` |
| **External functs.** | `malloc, free, write, va_start, va_arg, va_copy, va_end` |
| **Libft authorized** | `Yes` |
| **Description** | `Write a library that contains ft_printf(), a function that will mimic the original printf()` |

# Gamification



Problem-Based Learning

Gamification

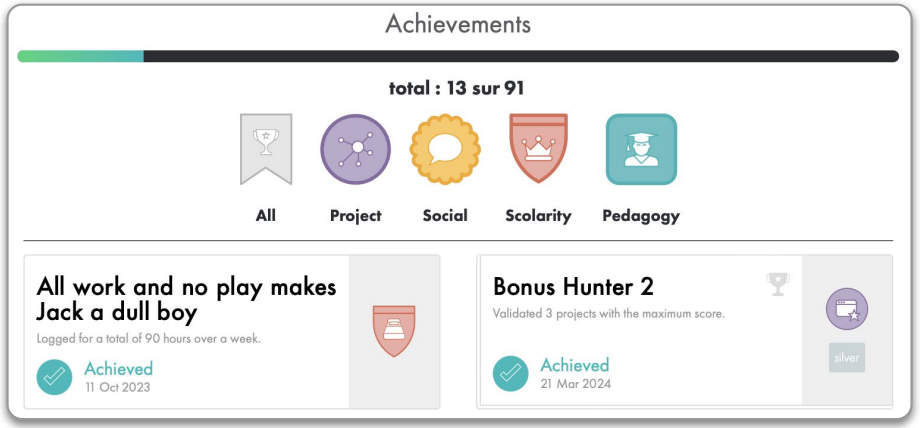
Peer Pedagogy

Community Development

# Gamification

- Reward system based on **experience points**, **user levels**, **achievements** and **titles**
  - ➢ earn experience by validating projects and scale levels
  - ➢ get badges or titles by completing missions

# Gamification



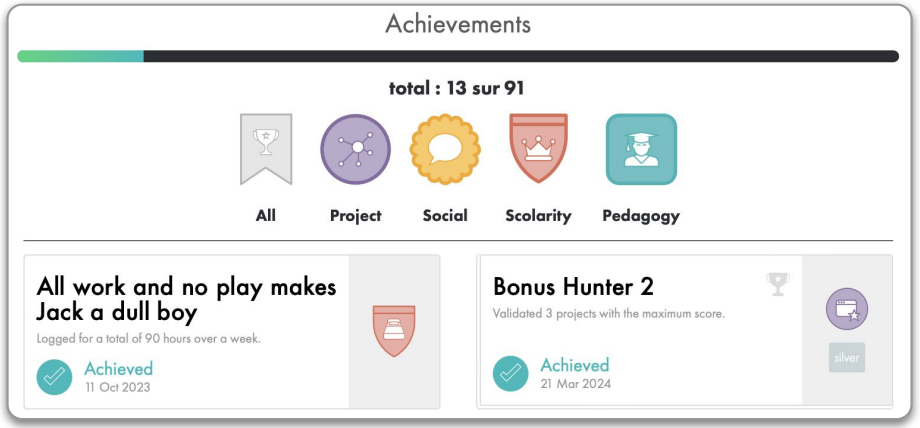
# Gamification

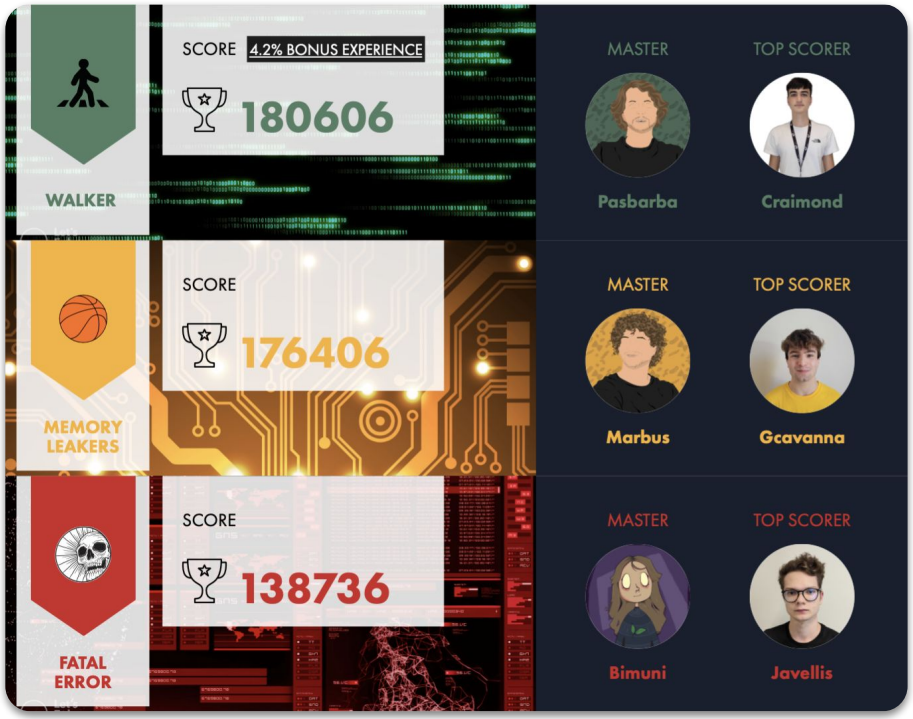
- Reward system based on **experience points**, **user levels**, **achievements** and **titles**
  - ➢ earn experience by validating projects and scale levels
  - ➢ get badges or titles by completing missions
- **Currency system** for evaluations
  - ➢ gain currency by evaluating others projects
  - ➢ spend currency to evaluate your projects

# Gamification

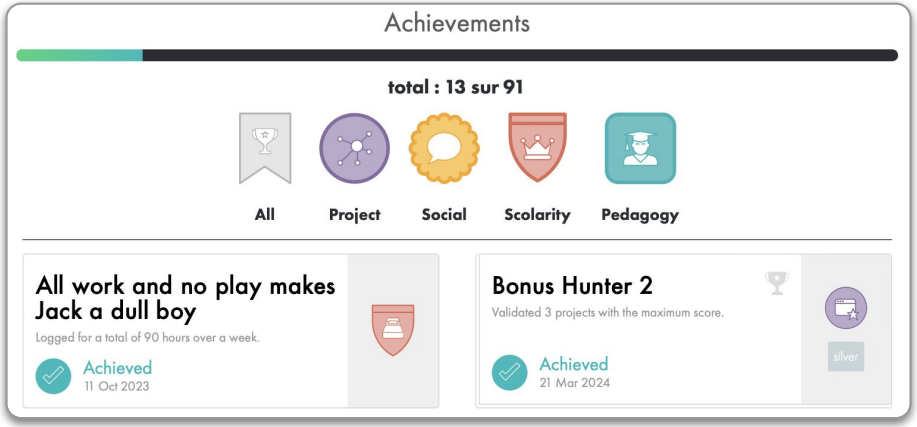
- Reward system based on **experience points**, **user levels**, **achievements** and **titles**
  - ➢ earn experience by validating projects and scale levels
  - ➢ get badges or titles by completing missions
- **Currency system** for evaluations
  - ➢ gain currency by evaluating others projects
  - ➢ spend currency to evaluate your projects
- Annual **coalitions league**
  - ➢ members of leading coalition gets extra experience points

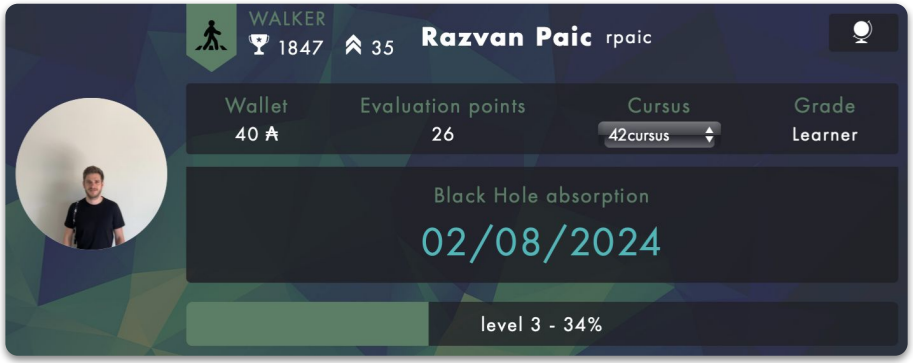
# Gamification

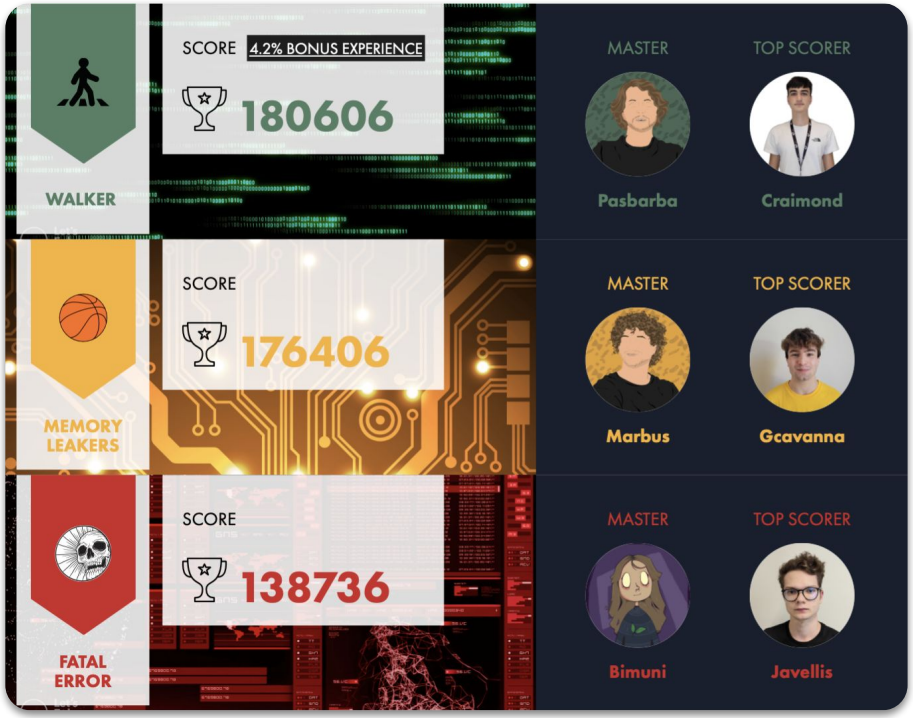


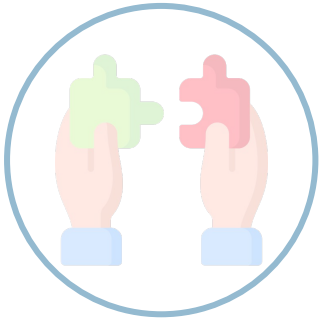
# Gamification



# Peer Pedagogy



Problem-Based Learning

Gamification

Peer Pedagogy

Community Development

# Peer Pedagogy

- **Collaboration as a key**: often the only way to overcome the greatest difficulties

  *e.g. “Got a question? Ask your peer on the right. Otherwise, try your peer on the left.”*

# Peer Pedagogy

*Photos from Florence campus*

# Peer Pedagogy

- **Collaboration as a key**: often the only way to overcome the greatest difficulties

  *e.g. "Got a question? Ask your peer on the right. Otherwise, try your peer on the left."*

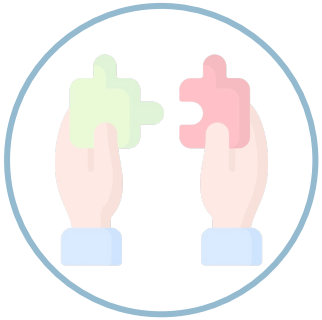
- **Information sharing**: advanced student can show the way for novices

  *e.g. "**Idea Exchange**: Engage in discussions about ideas, hypotheses, and solutions. Share thoughts on product quality, factors for success, and failure."*

- **Peer-reviews**: validate projects by addressing the absence of teachers

  *e.g. "Both parties should leave the defense with a sense of having learned something new, whether it's technical, relational, or organizational."*

# Community Development



Problem-Based Learning

Gamification

Peer Pedagogy

Community Development

# Community Development

- **Social events**: to tie the community together
  - ➢ “Happy Fridays”: students workshops, role-play games, …
  - ➢ Christmas party: buffet dinner, bingo, …
- **Network building**: to enforce collaboration
  - ➢ Group coding challenges
- **Community spaces**: to recover mental energy and overcome difficult moments together
  - ➢ Relaxation room: books, piano, …
  - ➢ Leisure room: table football, ping-pong, board games, …

# Community Development



*Photos from Florence campus*

# Research Questions

**$RQ_1$** *How does the 42's educational model contribute to the methods and practices of programming education?*

# Research Questions

**$RQ_1$** *How does the 42's educational model contribute to the methods and practices of programming education?*

**$RQ_{1.1}$** *How does **problem-based learning** impact programming education?*

**$RQ_{1.2}$** *How does **gamification** impact programming education?*

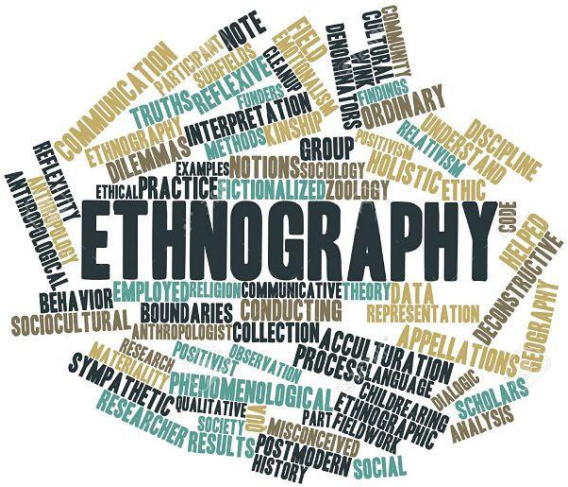
**$RQ_{1.3}$** *How does **peer pedagogy** impact programming education?*

**$RQ_{1.4}$** *How does **community development** impact programming education?*

# Research methodology

**What** is an ethnography:

- **qualitative inquiry**
- **deep immersion within the community** as part of it (extensive fieldwork)
- **first-hand access to crucial experiences** for analysis and interpretation



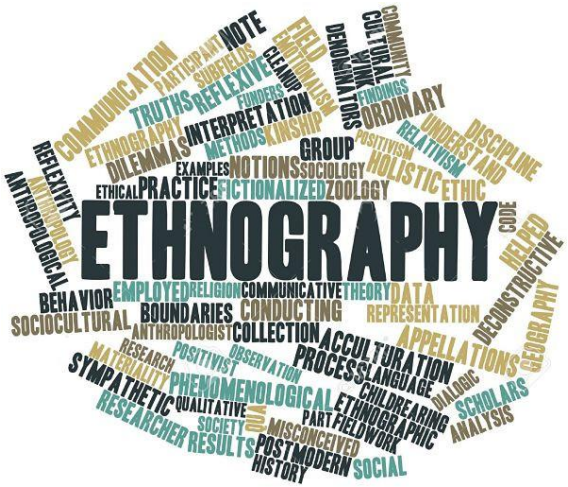
# Research methodology

**What** is an ethnography:

- **qualitative inquiry**
- **deep immersion within the community** as part of it (extensive fieldwork)
- **first-hand access to crucial experiences** for analysis and interpretation

**Why** an ethnography:

- necessity to **address the complexity of the method**
- **insights hidden in non-written patterns of daily life**
- highly descriptive **insider perspective**



# Ethnography

**DESIGN PHASE**

- **Participation level**: *Participant observation approach*
  - **Students point of view** with researchers eyes
  - **Equal footing** with students, but remaining **transparent** with them.
- **Duration**: ~6 months
- **Space and location**: *on-site* campus activities
- **Intent**:
  - Comprehensively understanding of the **pedagogical approach**
  - Unveiling its **actual impact on students learning**

**EXECUTION PHASE**

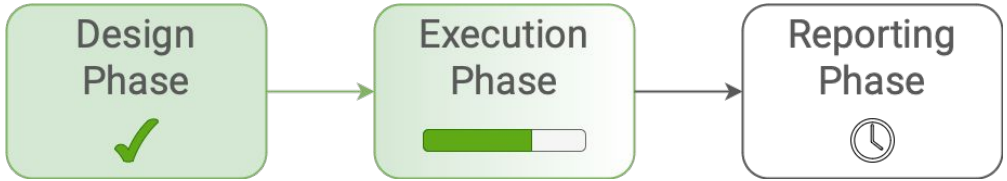
- **Modality**: attendance on campus (~3 days per week)
  - **carrying out exercises** as students
  - engaging in **unstructured and informal interviews**

# Limitations

- **Researchers background** and pre-built experience
  - **restriction to tools and knowledge to exploit** during students activities engagement
  - *bracketing* strategy to **address subjective biases being aware of our background**
- **Students biases** towards researchers
  - **informal interactions** to establish peer-like relationships
  - **interest and enthusiasm** on the part of students to participate

# PRELIMINARY RESULTS

# PRELIMINARY RESULTS



# Problem-Based Learning & Gamification

- Inductive approach:
  - **finding a solution** skill 🙂
  - **enforcing and reusing prior knowledge** 🙂
  - **learning at your own pace** 🙂
- Feedback and reward:
  - **gaining confidence** by submitting work 🙂
  - **triggering gratification** through points and levels 🙂
  - **balancing give-take phenomena** 🙂

➔ Student perception:

- **highly challenging and engaging approach**
- **discouragement mitigated by logical progression**
- **some time needed to adapt**

# Peer Pedagogy

- Cooperative learning:
  - **spontaneous collaboration** to overcome difficulties 🙂
  - **better comprehension of practical aspect** while comparing progress 🙂
  - **rigorous peer-review thanks to detailed instruction** and staff checks 🙂😕

➔ Student perception:
- **sensation of not being alone**
- **human contact** mitigates difficulties
- **validation of own acquired knowledge** by sharing solutions

# Community Development

- Social activities:
  - **promotion of social interactions** 🙂
  - **development of soft skills** and collaterals 🙂
  - **progress deceleration** due to excess of recreational aspects ☹️

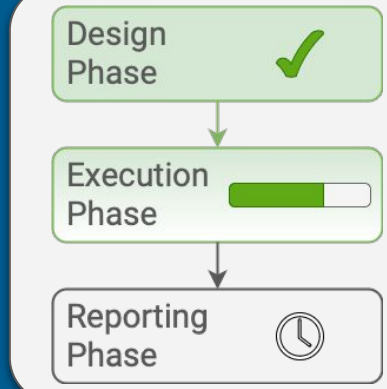
➔ Student perception:
- **sense of belonging**
- **social hub** to expand social circles
- **relaxation opportunities** during hard periods

# Learning Programming without Teachers: An Ongoing Ethnographic Study at 42

## Summary


**$RQ_1$**: *How does the 42's educational model contribute to the methods and practices of programming education?*

**$RQ_{1.1}$**: *Problem-Based Learning*
**$RQ_{1.2}$**: *Gamification*
**$RQ_{1.3}$**: *Peer Pedagogy*
**$RQ_{1.4}$**: *Community Development*




Problem-Based Learning ☺
Gamification ☺
Peer Pedagogy ☺😐
Community Development ☺☹

→ *such a pedagogical method with many freedoms requires a lot of responsibility to fully reap all its benefits*

## Thank you, and… it's Q&A time!

**Kevin Maggi**
University of Florence
*kevin.maggi@unifi.it*

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE