NTNU
Kunnskap for en bedre verden

# Code rotation

CoPCSE, Oppdal, 2018-11-27

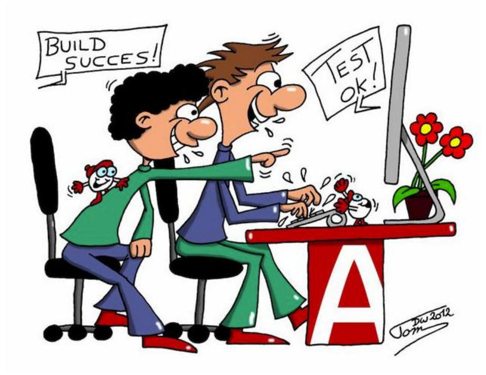
Girts Strazdins, gist@ntnu.no, NTNU (Ålesund)

Three groups work on same programming task...

When they are done, they rotate code...

Task 1,
Implementation B

Task 1,
Implementation A

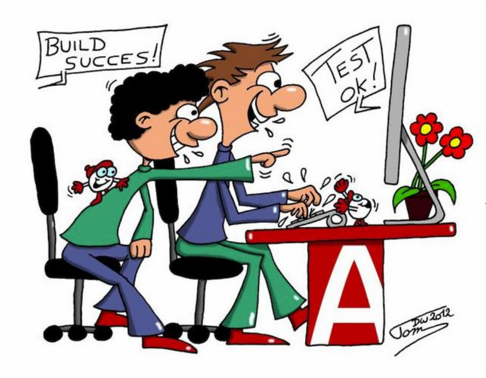


Task 1,
Implementation C

Tehn they continue the same for the next task...

Task 1 Implementation A
+ Task 2 Implementation B

Task 1 Implementation C
+ Task 2 Implementation A



Task 1 Implementation B
+ Task 2 Implementation C

## Resulting code after x steps

- 3 projects, with all x steps implemented
- Every team has worked on each project
- Each project has all three teams as contributors

## Motivation (hope)

1. Teams learn from each other
2. Responsibility
3. Realistic setting
4. Side-goal: GIT, level 2+

# Trial in "Nettverksprogrammering"

- Programming task: TCP client for a chat, 8 steps
- ~70 students, 1-3 in each team. Data+Automasjon.
- Demo for teacher at the end
- GIT with branch for each step, each team
  - Template in Git Classroom

| | **Which team implements which steps for which project** | | | |
|---|---|---|---|---|
| | **Steps 1+2 in** | **Steps 3+4 in** | **Steps 5+6 in** | **Steps 7+8 in** |
| **Team 1** | Project 1 | Project 3 | Project 2 | Project 1 |
| **Team 2** | Project 2 | Project 1 | Project 3 | Project 2 |
| **Team 3** | Project 3 | Project 2 | Project 1 | Project 3 |

## Evaluation

- Qualitative discussion with team during demo
- Questions (approximate):
  - How did the project go?
  - What was most difficult?
  - Did you learn something from code rotation?
  - How did you synchronize among teams?
  - Are you comfortable with GIT?

## Conclusion #1

**This approach is good for learning GIT**

- Answers for "Are you comfortable with GIT now?"
  - Yes (more or less)
  - No, but much better than I was before the course

- Few say "don't understand it"
  - Most of those failed in OOP course

## Conclusion #2

**Tasks should be larger and allow improvisation.**

Answers for "Did you learn from others' code?":

- It was the same as mine
- Few say: I got surprised by code block X, had to ask

## Conclusion #3

**Realistic setting and teamwork experience.**

- Most worked together
- When asked "was GIT branching hard?" Some students say "Yes, but we should learn it, because that is what industry uses".

## Conclusion #4

**Hard to conclude about responsibility.**

- Risk for (unnecessary) peer pressure and criticism?

- Some say: *they* delayed, I had to work on their part
- The "usual struggle" – assignment was easy for some, hard for some. Ok for most.
- Almost everyone said: hard to begin, easy at the end.
  - Is that good or bad?

# Summarized conclusions

1. Teams learn from each other  - unsure
2. Responsibility – not really?
3. Realistic setting - yes
4. Side-goal: GIT, level 2 - yes

## Challenge

- What to do with students who:
  - Failed OOP?
  - Study automation and ask "Why do we need this?"
- Too much *on the plate* = zero learning?
- Adding GIT, branches, rotation makes it worse?