Pair Programming in Physics courses

29 Jan 2018 – Edited 18 June 2019

Michelle Kuchera

Davidson College

What is Pair Programming?

- Collaborative learning method in which students program in pairs instead of individually.
- Improves college students' programming competency and increases the likelihood that both male and female students become and remain computer science majors.
- Students work in tandem at one computer while completing regular programming assignments. The "driver "controls the mouse and keyboard while the "navigator" makes suggestions, points out errors, and asks questions.
- The partners routinely switch roles to gain the benefits of each role.

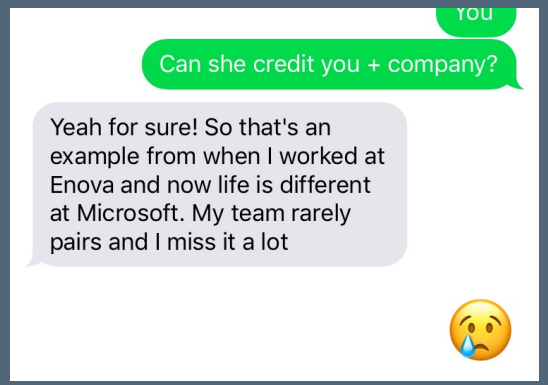
https://www.ncwit.org/resources/pair-programming-box-power-collaborative-learning

Some research results

- **Using collaborative learning research to enhance pair programming pedagogy**
  https://dl.acm.org/citation.cfm?id=1113381
- **Undergraduate student perceptions of pair programming and agile software methodologies: verifying a model of social interaction**
  https://collaboration.csc.ncsu.edu/laurie/Papers/Agile2005.pdf

- Also, pair programming exists in the workplace for many of the same reasons.

Pair programming in the workplace



- Pair programming has been one of the most useful tools to keep our projects going. We usually have the least experienced dev be the driver (person typing). Least experienced can mean the most junior or less familiar with the code base. Both engineers look at the same screen and take turns typing if one person is losing focus or getting tired. While counterintuitive, having two engineers working on the same task helps keep the focus on the problem at hand and avoid slowdowns to check for syntax and other minor validations. Pair coding also encourages relevant knowledge sharing. A few months ago my team was developing a service to aggregate credit reports. I had a big task validating requests from the clients and packaging responses from third-party data vendors. The validations required the use of regular expressions. My coworker and I paired to make sure edge cases for the regular expression were appropriately handled. Countless times one person would remember something the other wouldn't or share an IDE tip that we didn't know about. In one day we not only finished the task but we pretty much developed the backbone of the whole service. After that, we broke up to write unit and end to end tests. The pairing sessions made the entire process much smoother and we finished well ahead of schedule. - Deliana at Enova

Pair programming in the workplace

- The best pairing experience I had was when I first started. We had two keyboards, two mice, one display. We took turns "driving." As you write code, you have to sort of narrate your thoughts so the other person can follow your thought process. That alone is a skill in itself.

  Usually we did the sort of "driver/navigator" interaction, but sometimes we would do a different one called "ping pong" where one person (usually the more senior engineer) writes a failing test that captures the needed functionality, and then the other makes the test green. That goes hand in hand with TDD.

  The last thing that was really helpful was using the pomodoro method while pairing. Pairing is really intense, and doing 25 minutes on, 5 minutes off is helpful to keep you fresh and gives you a chance to use the bathroom or check your phone.

- -Craig at Ultimate Software

Pair programming in the workplace

- We don't officially pair program, but many times on things we want to make sure more than one person knows about early on before we have a chance to document the solution, we'll pair up. The person not 'coding' will often act as a researcher looking up things and proposing ideas while the person coding is manipulating data, testing, writing code.
- -Geo at Ruvos

- Very similar to our active learning methods from physics education research.
- In class, workshop style:
  - Randomly assign partners (not pedagogically backed).
  - Run script to switch who is “driving” at given intervals of time.
  - Green/red post-its indicate who needs help and who is done.
  - Encourage students who are done to help those that need it.

- Often working through Jupyter notebooks with explicit instructions.
- In my classes:
  - Physics classes: pair program in the classroom or lab.
  - Intro classes: NO programming outside of the classroom or solo
  - Upper-level: at-home coding consistent with homework rules.
  - CS classes – pair program for ALL work, in class and homework.
  - Pair programming often preceded by whiteboard activities.

Observations

- Students overall appreciate and enjoy the in-class pair programming.
- Learning in the class is evident.
- Solo programming would be hard to support in the classroom.
- Gives students confidence for solo work or homework.
- Eases difference in coding abilities.

https://github.com/mpkuchera/Oslo2019