Lecture XIII - Presentations and Discussion

Programming with Python

Dr. Tobias Vlćek

EP 13: Demo Day

The investor is in the room

Back in Session VI she stood at a whiteboard and asked why everything here was built from scratch. Today the answer is on your screens: built by you, with the standard library, pandas, plots, git, and an agent you learned to supervise.

The rules: **7 minutes** presenting, **3 minutes** of questions, order drawn at random.

While you watch

- Note one thing you would have done differently, and one thing you are taking for your own next project
- Questions are part of the format: specific, kind, useful

Your Future with AI

AI and your first job

- Thousands of developers: **+26%** tasks done with AI, **juniors gained most** [1]
- Experienced developers on their own code: **19% slower**, while *feeling* 20% faster [2]
- Learners with AI: **50%** on a mastery quiz vs **67%** without, worst on **debugging** [3]
- The fix: hints instead of finished solutions erased the harm [4]. Part I was exactly that

What stays valuable

- The verification boundary: usable delegation ends where your ability to **check** ends. Growing that boundary outward is your career
- **Comprehension debt**: code that works today but nobody understands tomorrow. The interest comes due when a reviewer asks “why does this work?”
- Your edge, deliberately kept: **explain every line before you accept it**, and code without AI sometimes

Discussion: your future with AI

> **Note**
> Three questions, five minutes, show of hands:
>
> - Which task from the past weeks would you **never** delegate to an AI?
> - What will an entry-level logistics analyst do in 2030, when AI drafts the first version of everything?
> - If AI writes most junior code, where do the senior people come from?

Send-off

Keep programming

The skill fades if it sits idle. A few ways to keep it alive:

- Use Python in your **thesis**: data cleaning, analysis, plots you can defend
- Find a way to **apply it at work**: the repetitive task nobody wants to do by hand
- **Advent of Code**: free, ad-free programming puzzles, one a day from **December 1st**. A genuinely fun way to keep the muscle warm.

Thank you

- We covered the basics of Python: from a first `print` to a real project on your own machine
- You debugged, you verified AI, you shipped real numbers
- I hope you enjoyed it, and I hope it's useful long after the grade
- If you have questions or feedback, please tell me

All the best for your studies and your career. Now go build the thing.

Literature

Keep reading

The literature list of this course has more books, several of them free online.

References

Bibliography

[1] K. Z. Cui, M. Demirer, S. Jaffe, L. Musolff, S. Peng, and T. Salz, "The Effects of Generative AI on High-Skilled Work: Evidence from Three Field Experiments with

Software Developers," *Management Science*, p. mnsc.2025.00535, Feb. 2026, doi: 10.1287/mnsc.2025.00535.

[2] J. Becker, N. Rush, E. Barnes, and D. Rein, "Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity." Accessed: Oct. 26, 2026. [Online]. Available: https://arxiv.org/abs/2507.09089

[3] J. H. Shen and A. Tamkin, "How AI Impacts Skill Formation." Accessed: Oct. 26, 2026. [Online]. Available: https://arxiv.org/abs/2601.20245

[4] H. Bastani, O. Bastani, A. Sungu, H. Ge, Ö. Kabakcı, and R. Mariman, "Generative AI without Guardrails Can Harm Learning: Evidence from High School Mathematics," *Proceedings of the National Academy of Sciences*, vol. 122, no. 26, p. e2422633122, Jul. 2025, doi: 10.1073/pnas.2422633122.