Course Overview

Week 1

The future belongs to those who believe in the beauty of their dreams.

— Eleanor Roosevelt

Synopsis

- Expose students to mobile development, software design issues, processes and project management techniques

- Allow students to experience software development and project management by working in a team to develop a mobile application

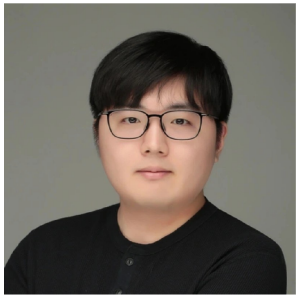
Instructor

- Name: Youngki Lee (이영기)
- Office: 301-413
- Office Hour: Wed. 11AM-12PM
- Phone : 02-880-1726
- E-mail: youngkilee@snu.ac.kr
- Research Homepage: http://hcs.snu.ac.kr

Teaching Assistants

Sangwon Park

Taeho Kang



Hyunsoo Kim

Jaeyeon Park

Sieun Park

How to Contact TAs?

- Email: ta_swpp@hcs.snu.ac.kr
  - For personal or private issues (e.g., attendance related)
  - Title format: **[SWPP2025] title** (Ex: [SWPP2025] 결석계 제출)
  - Please include your **name** and **student ID**
  - Requests to personal emails will not be attended
- Slack:
  - For lecture or project related issues
    - Check the guide on #announcement
    - Do NOT DM to TA in Slack
  - Slack invitation link
    - If the link doesn't work, send an email to the TAs

Protocol

- Respect their time
- Be punctual for meetings with them
- Be reasonable in requests
- They have been instructed to teach, not to give answers

Content

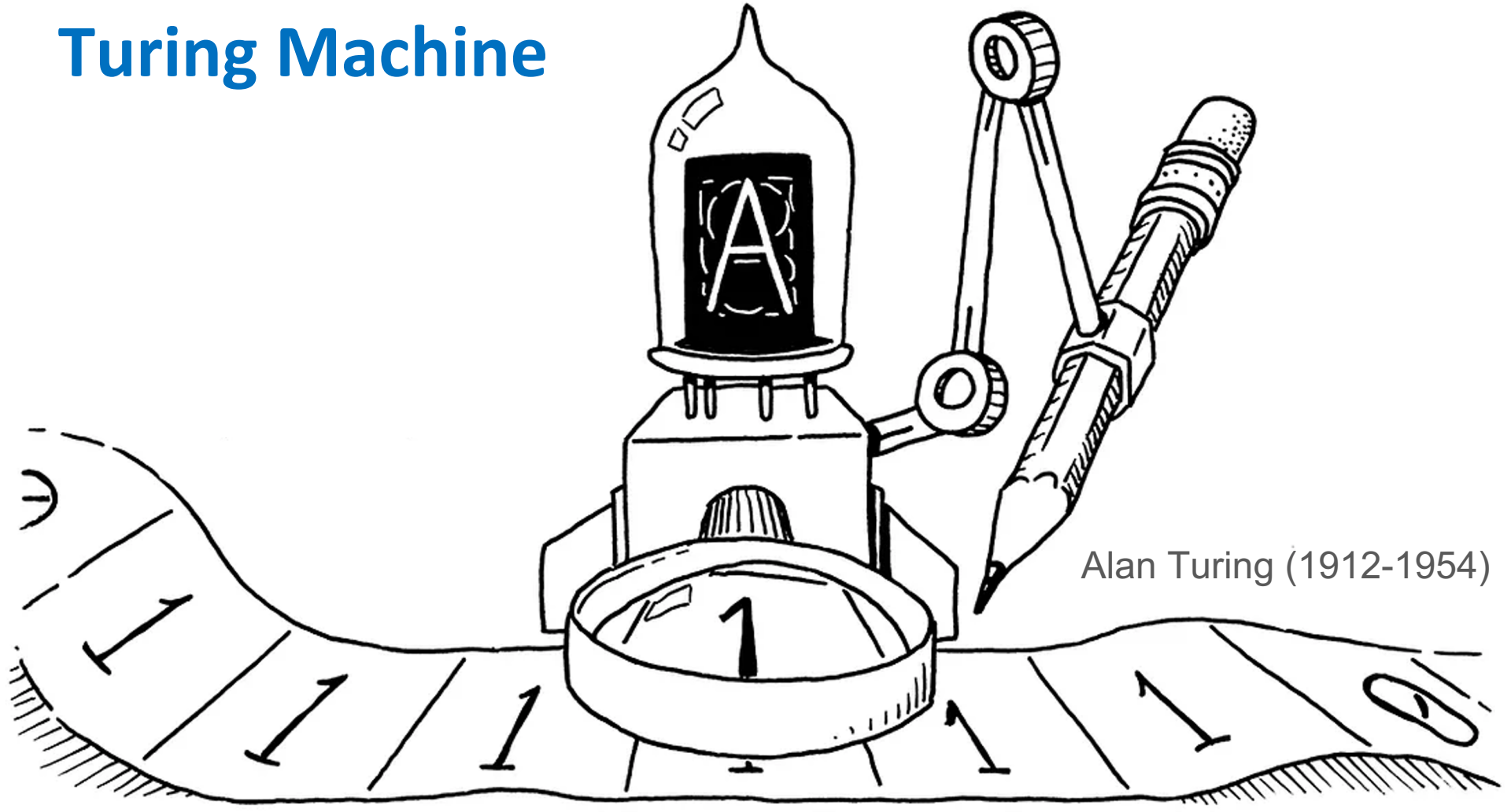
- What is Software Engineering?
- Course Overview

What is Software?

> Computer programs, procedures and possibly associated documentation and data pertaining to the operation of a computer system. Includes both executable and non-executable software, such as fonts, graphics, database records [...].
>
> *– ISO/IEC/IEEE 24765:2017(en) Systems and software engineering (Vocabulary)*

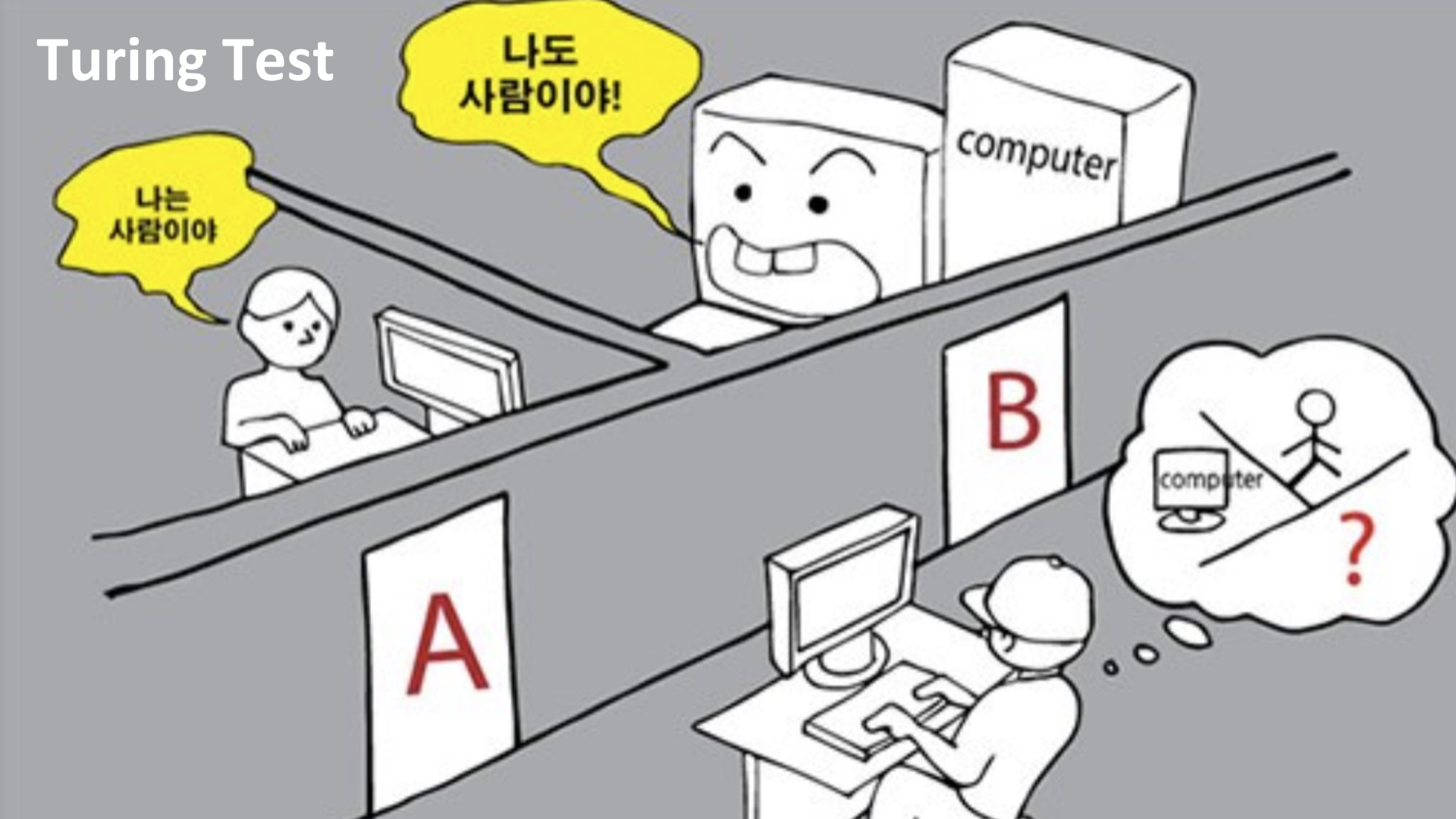
- Software encompasses everything that tells a computer how to perform specific tasks or functions. It is intangible and exists in digital form, typically stored on storage devices like hard drives or distributed over networks

Turing Machine



Alan Turing (1912-1954)

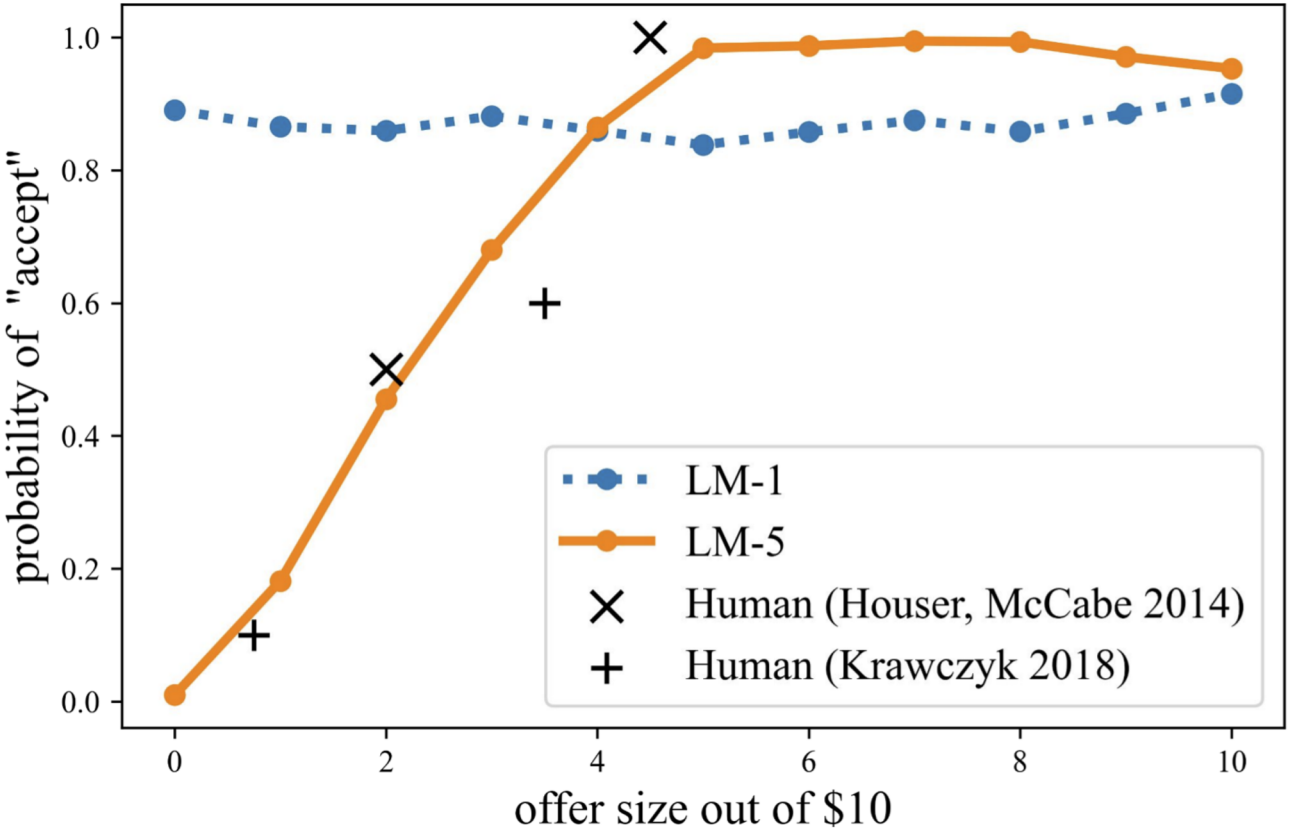
Turing Test



Ultimatum Game

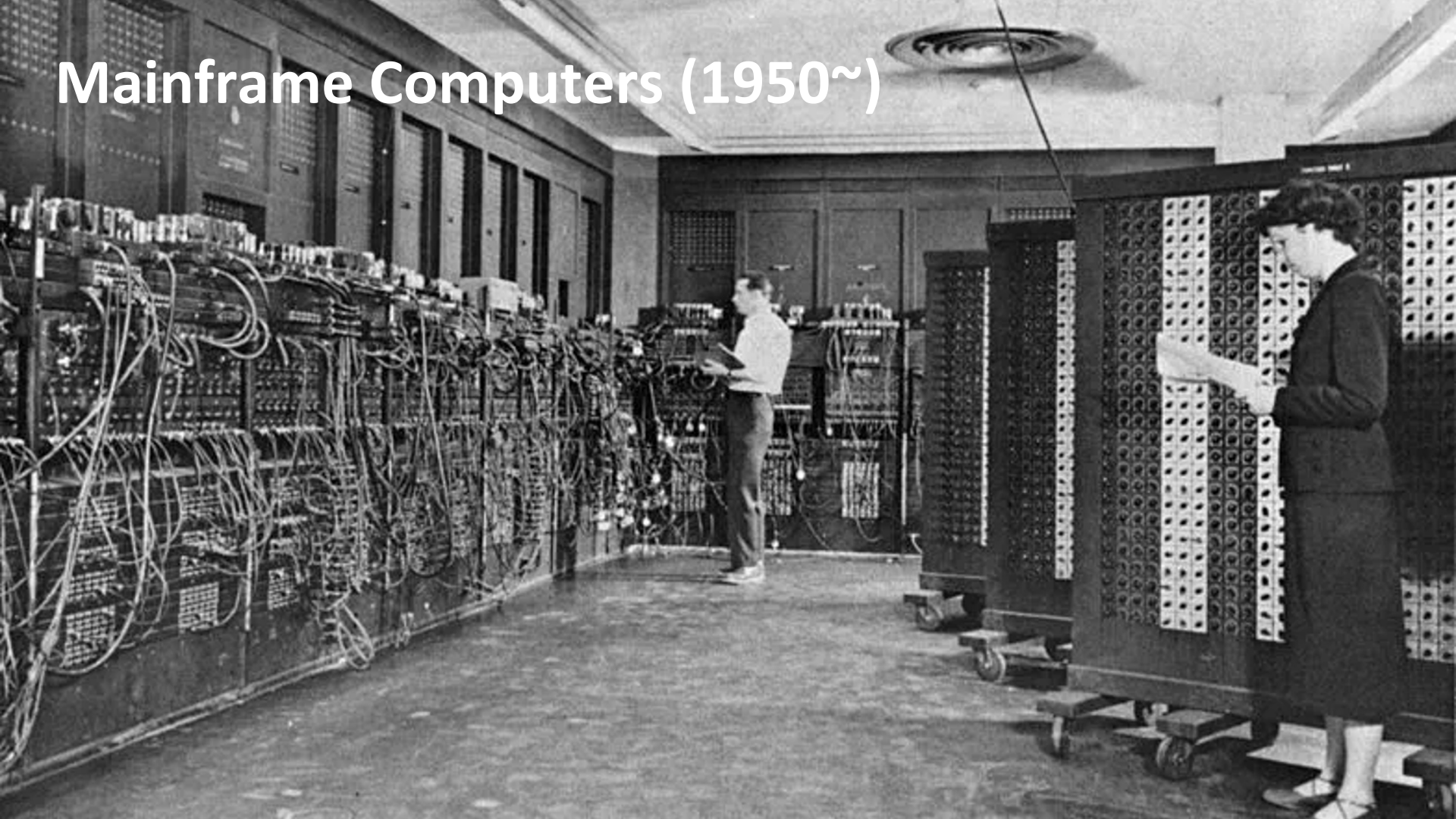
[ICML 2023] Using large language models to simulate multiple humans and replicate human subject studies.

AI Pass Turing Test?

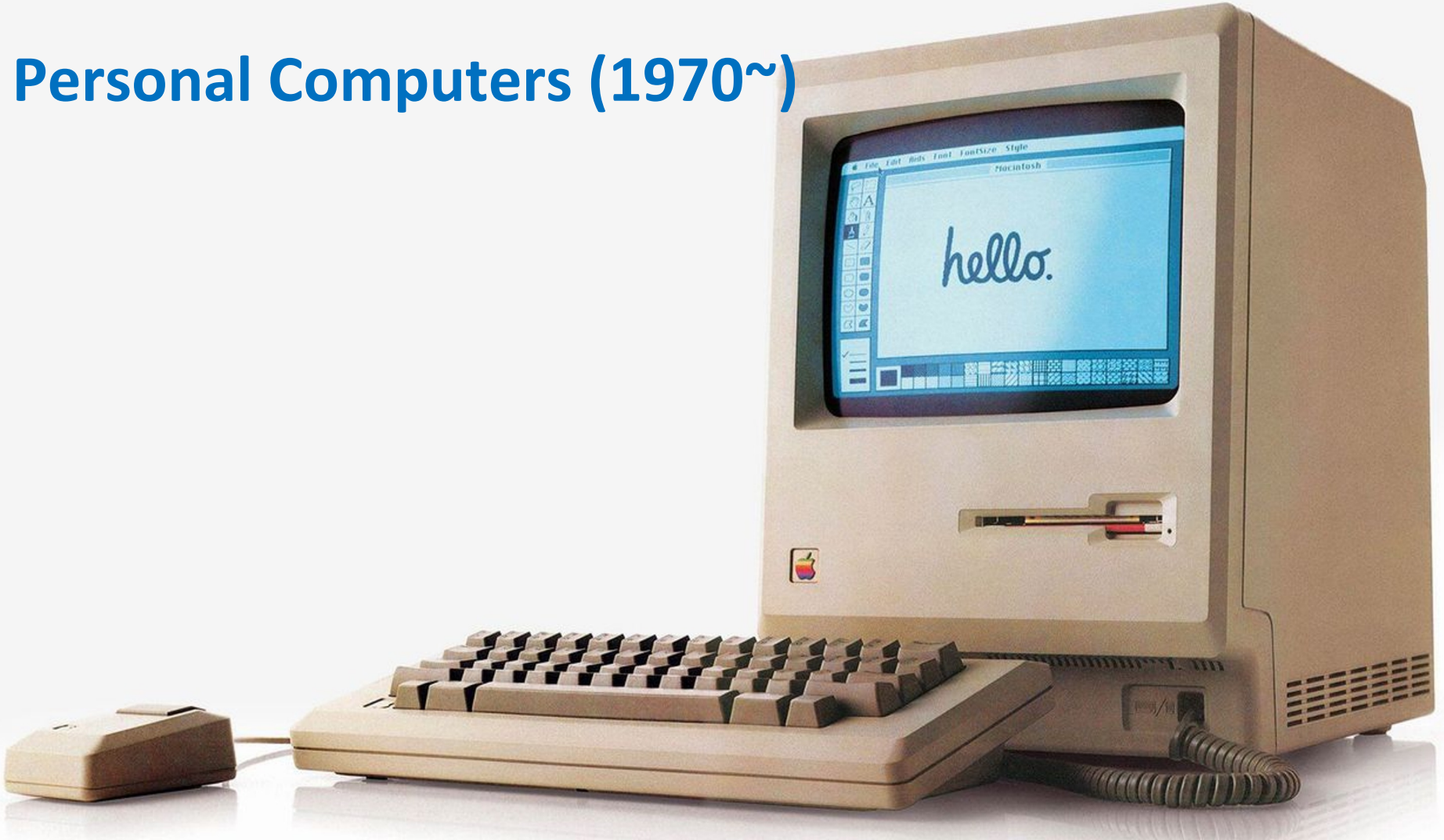


[ICML 2023] Using large language models to simulate multiple humans and replicate human subject studies.

Mainframe Computers (1950~)



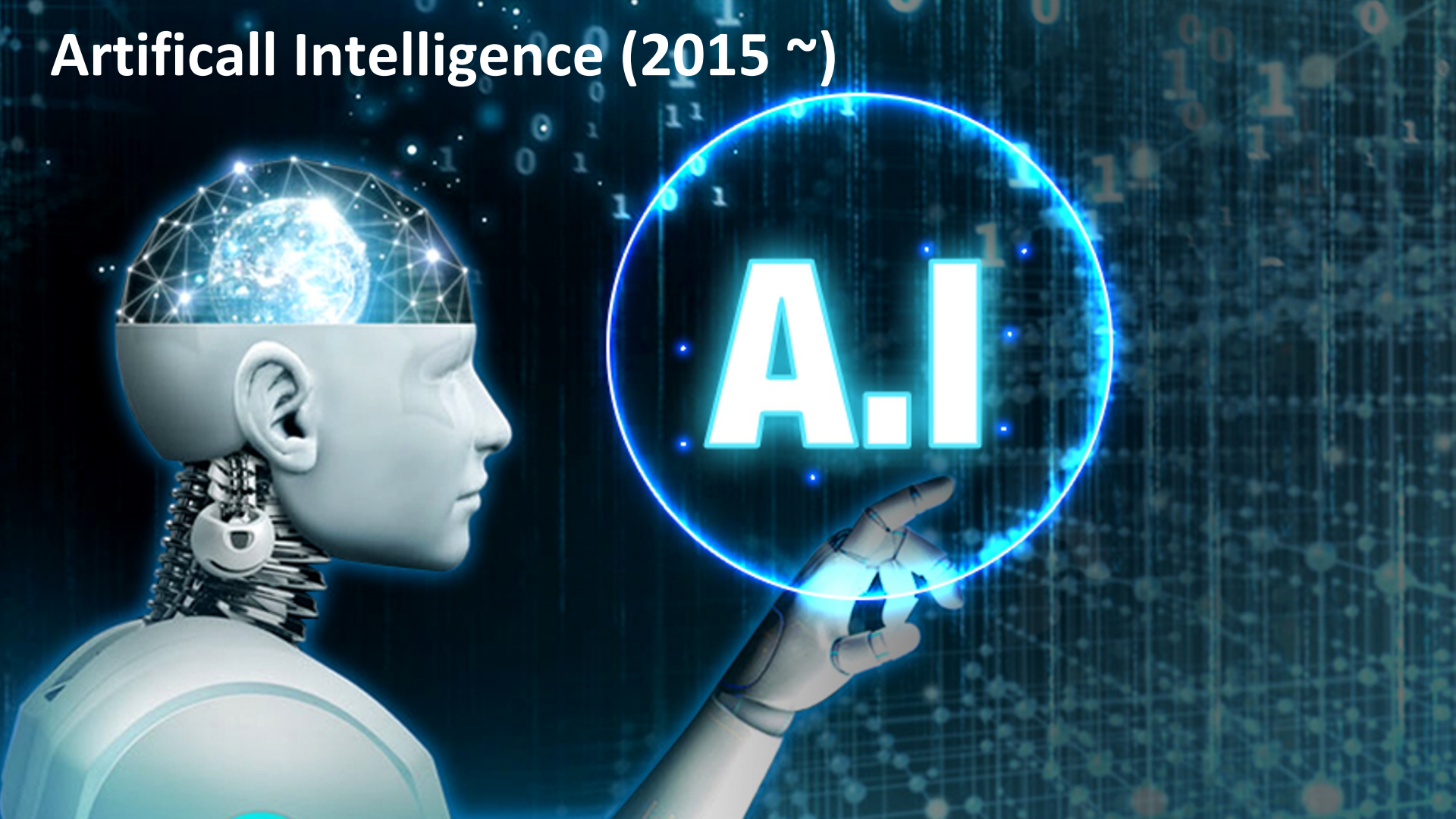
Personal Computers (1970~)



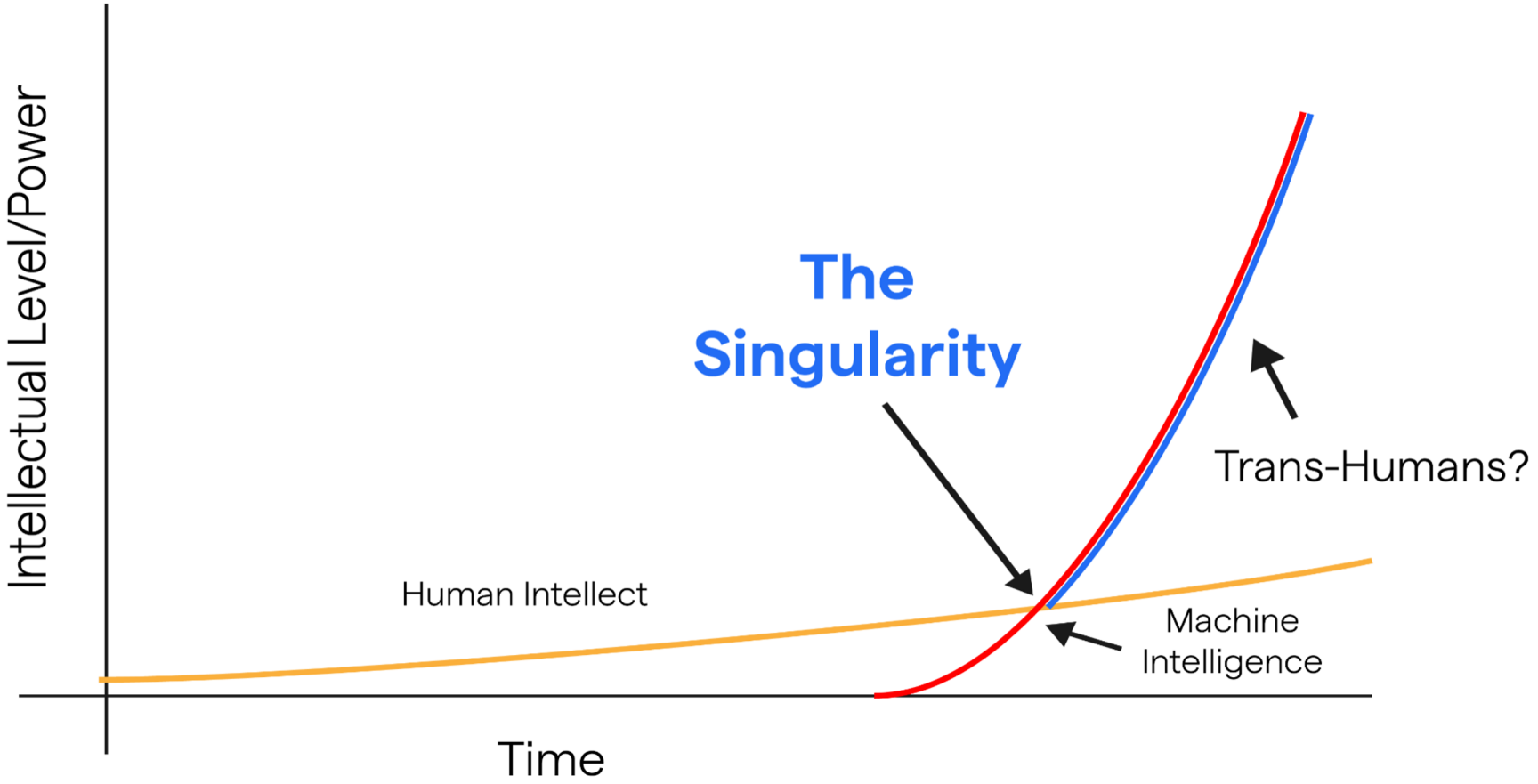
Internet and Web (1980~)

Mobile and Cluod (2000~)

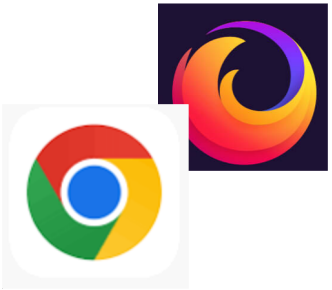
Artificall Intelligence (2015 ~)

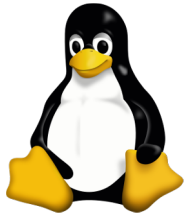


The Singularity

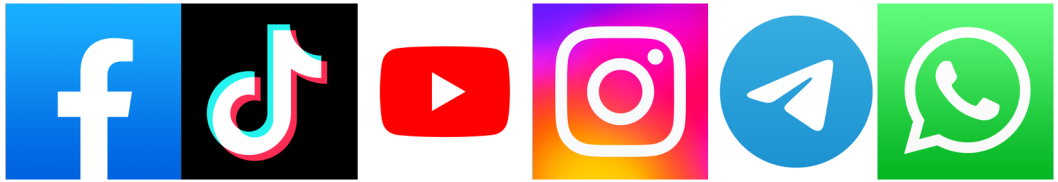


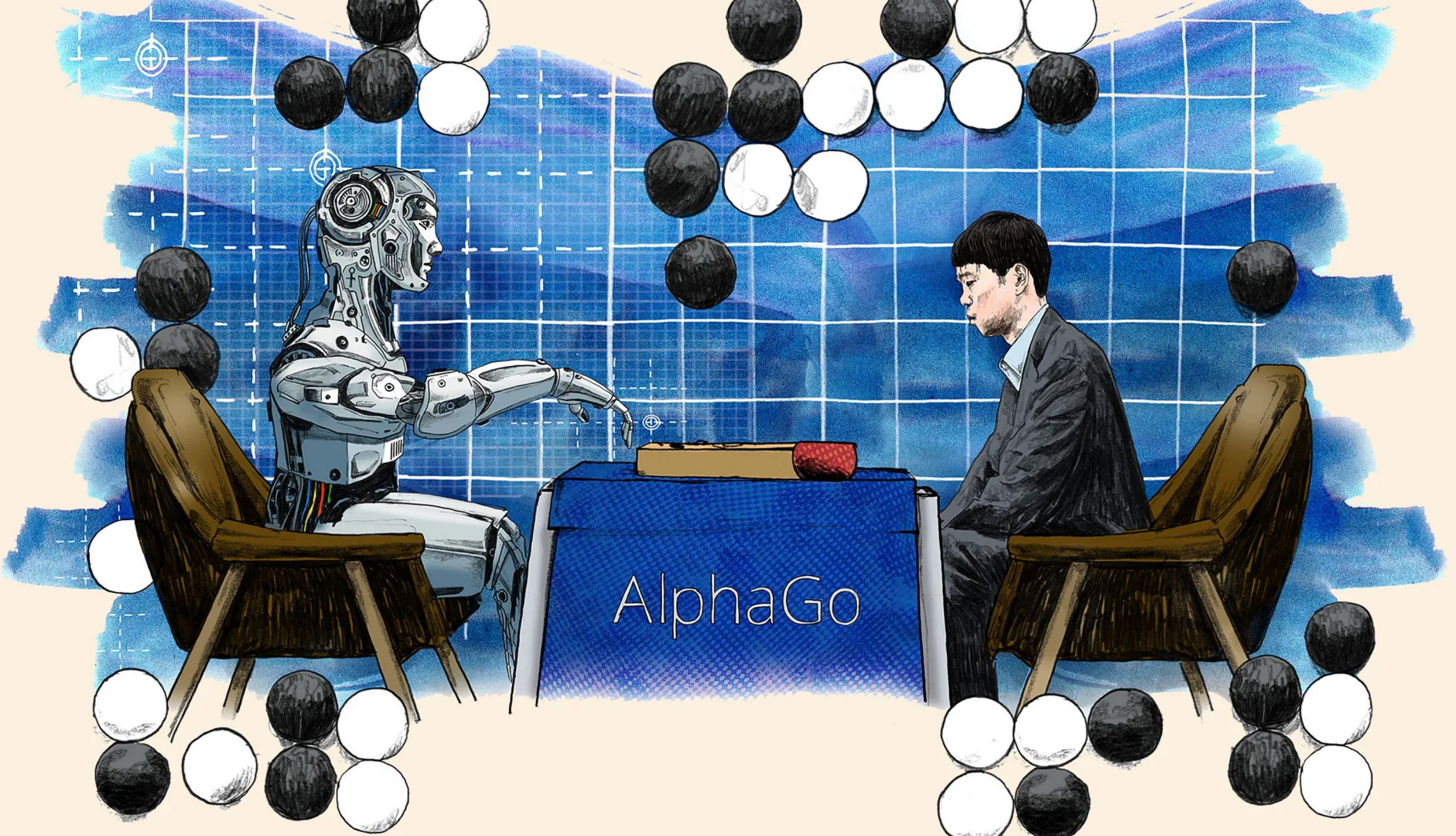
Conventional Software



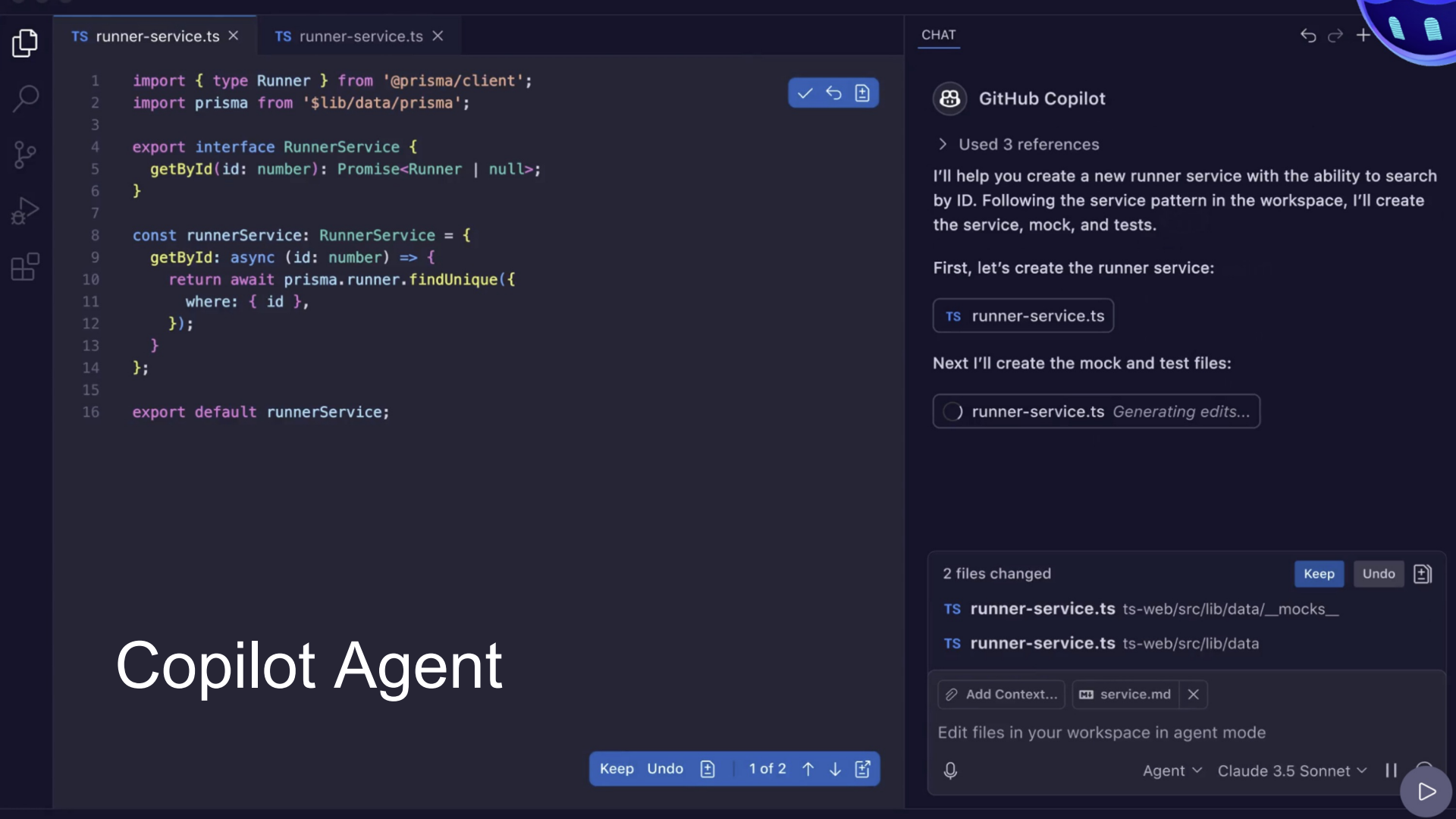


Conventional Software



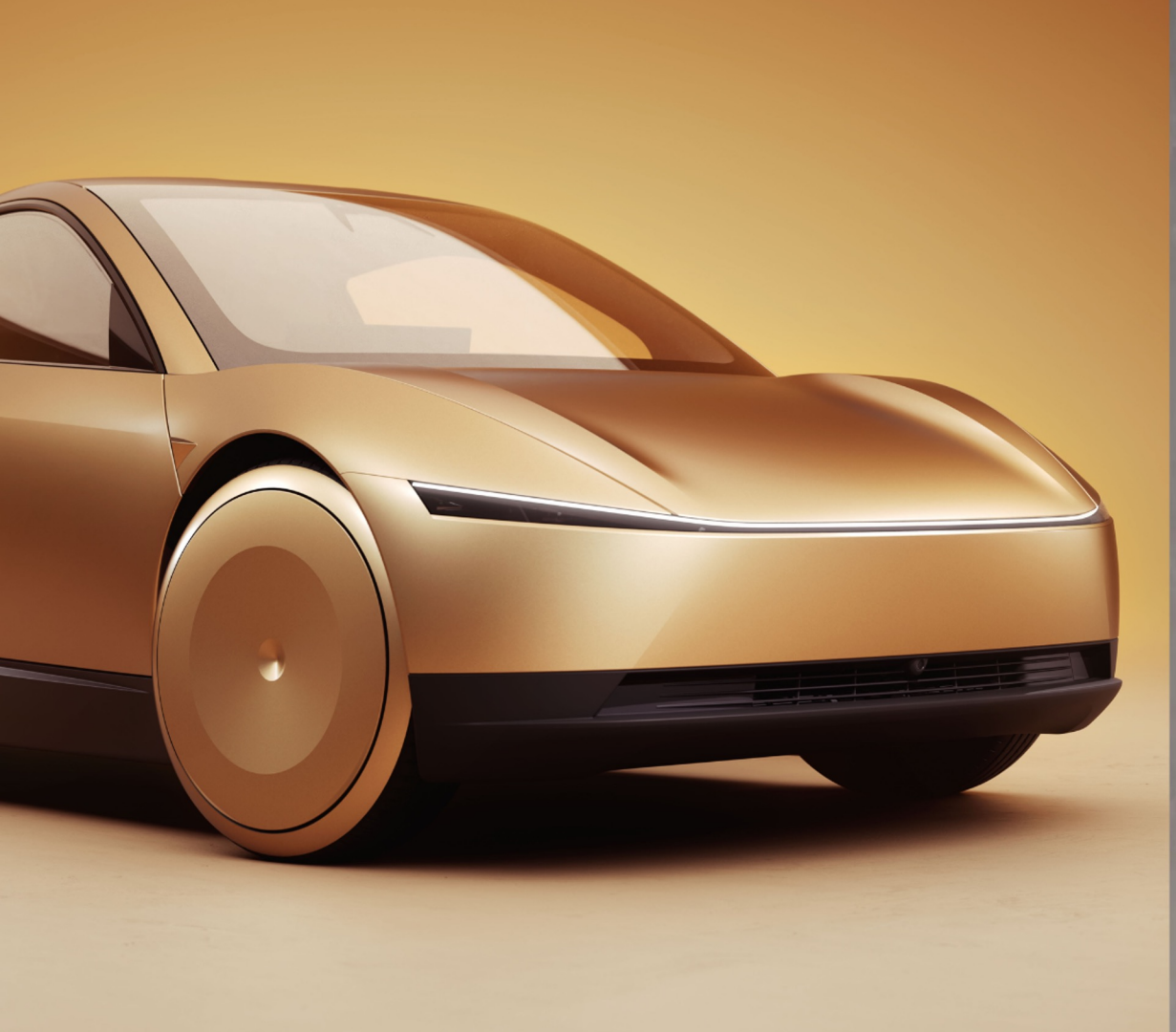

AlphaGo


ChatGPT


TS runner-service.ts
TS runner-service.ts
1 import { type Runner } from '@prisma/client';
2 import prisma from '$lib/data/prisma';
3
4 export interface RunnerService {
5 getById(id: number): Promise<Runner | null>;
6 }
7
8 const runnerService: RunnerService = {
9 getById: async (id: number) => {
10 return await prisma.runner.findUnique({
11 where: { id },
12 });
13 }
14 };
15
16 export default runnerService;
Keep Undo 1 of 2
CHAT
GitHub Copilot
Used 3 references
I'll help you create a new runner service with the ability to search by ID. Following the service pattern in the workspace, I'll create the service, mock, and tests.
First, let's create the runner service:
TS runner-service.ts
Next I'll create the mock and test files:
runner-service.ts Generating edits...
2 files changed
Keep
Undo
TS runner-service.ts ts-web/src/lib/data/__mocks__
TS runner-service.ts ts-web/src/lib/data
Add Context...
service.md
Edit files in your workspace in agent mode
Agent
Claude 3.5 Sonnet
Copilot Agent


AlphaFold 3


Credit: Meta




TESLA

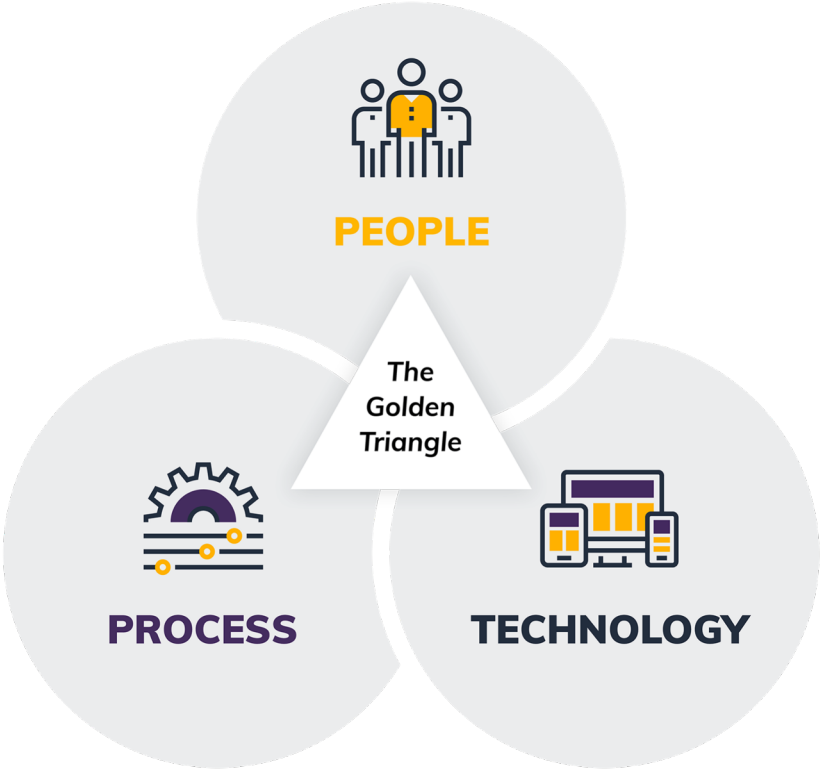
What is Software Engineering?

> Application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
>
> *— ISO/IEC TR 19759:2016, Software Engineering, Guide to the Software Engineering Body of Knowledge*

- Software engineers apply engineering principles and knowledge of programming languages to build software
- Software engineering deals with the design, development, testing, and maintenance of software applications
- It helps to build safe, secure, reliable, performant, scalable, and manageable software

Three Factors for Software Engineering

Note: Technology is just one aspect!



Why Learn Software Engineering?

- Learn how to build quality software as a team in a systematic and disciplined way

- Many of us have done small-scale software projects in terms of lines of code, number of users, team sizes, project duration, and budget.

- It is important to scale the knowhow for future med- to large-size real-world projects beyond the course projects

Poorly Engineered Software? (1/3)

Minor/major app crashes:

Outlook India

Instagram Suffers Global Outage As App Crashes And Glitches Frustrate Users

Users worldwide took to social media to express their dissatisfaction, while Instagram's official PR account on Twitter remains silent about...

1 month ago

The Guardian

Facebook outage: what went wrong and why did it take so long to fix after social platform went down?

Billions of users were unable to access Facebook, Instagram and WhatsApp for hours while the social media giant scrambled to restore...

Computer Weekly

Microsoft cloud users hit by global outage linked to Azure Active Directory issue

Microsoft claims to have resolved a global outage that left users of various services within its cloud portfolio unable to access the

Mar 16, 2021

The Guardian

https://www.theguardian.com › technology › 2012 › sep

Apple Maps service loses train stations, shrinks tower and ...

Thu 20 Sep 2012 08.00 EDT ... **Apple Maps** replaces Google Maps as the default map service on the latest version **Apple's iPhone** and iPad software.

Poorly Engineered Software? (2/3)

Expensive bugs:

Y2K problem:
Individual companies predicted the global damage caused by the bug would require anything between $400 - $600 billion to rectify

An electronic sign at École centrale de Nantes incorrectly displaying the year 1900 on 3 January 2000 (on the screen it translates to "Welcome to the Ecole Centrale de Nantes 12:09 p.m. January 3, 1900.")

Source: https://en.wikipedia.org/wiki/Year_2000_problem

Poorly Engineered Software? (3/3)

Fatal bugs:

Software bug lead to death in self-driving car:
Sensors detected Elaine Herzberg, but software reportedly decided to ignore her

NTSB officials inspecting the vehicle that killed Elaine Herzberg in a March crash in Arizona

Source: https://arstechnica.com/tech-policy/2018/05/report-software-bug-led-to-death-in-ubers-self-driving-crash/

Course Objectives

- Build novel and usable software
  - Learn mobile/AI software development skills
  - Learn programming practices and design concepts to develop quality software

- Experience the end-to-end process of developing software as a team

Course Connections (1/2)

- Connection with lower-level courses: You will learn how to apply the basic programming skills for real-world software projects
  - Computer Programming (컴퓨터 프로그래밍)
  - Computer Architecture (컴퓨터구조)
  - Data Structure (자료구조)
  - Algorithm (알고리즘)
  - System Programming (시스템 프로그래밍)

Course Connections (2/2)

- Connection with upper-level courses: You will build cornerstones to perform your future course projects in a more systematic and organized way
  - Mobile Computing and Applications (모바일 컴퓨팅과 응용)
  - Mobile and Ubiquitous Computing (모바일 및 유비쿼터스컴퓨팅)
  - Creative Integrated Design (창의적 통합 설계)
  - Any other courses where you manage team projects.

Logistics: Classes

- Lecture
  - Tuesday 14:00 ~ 15:15
  - Thursday 14:00 ~ 15:15

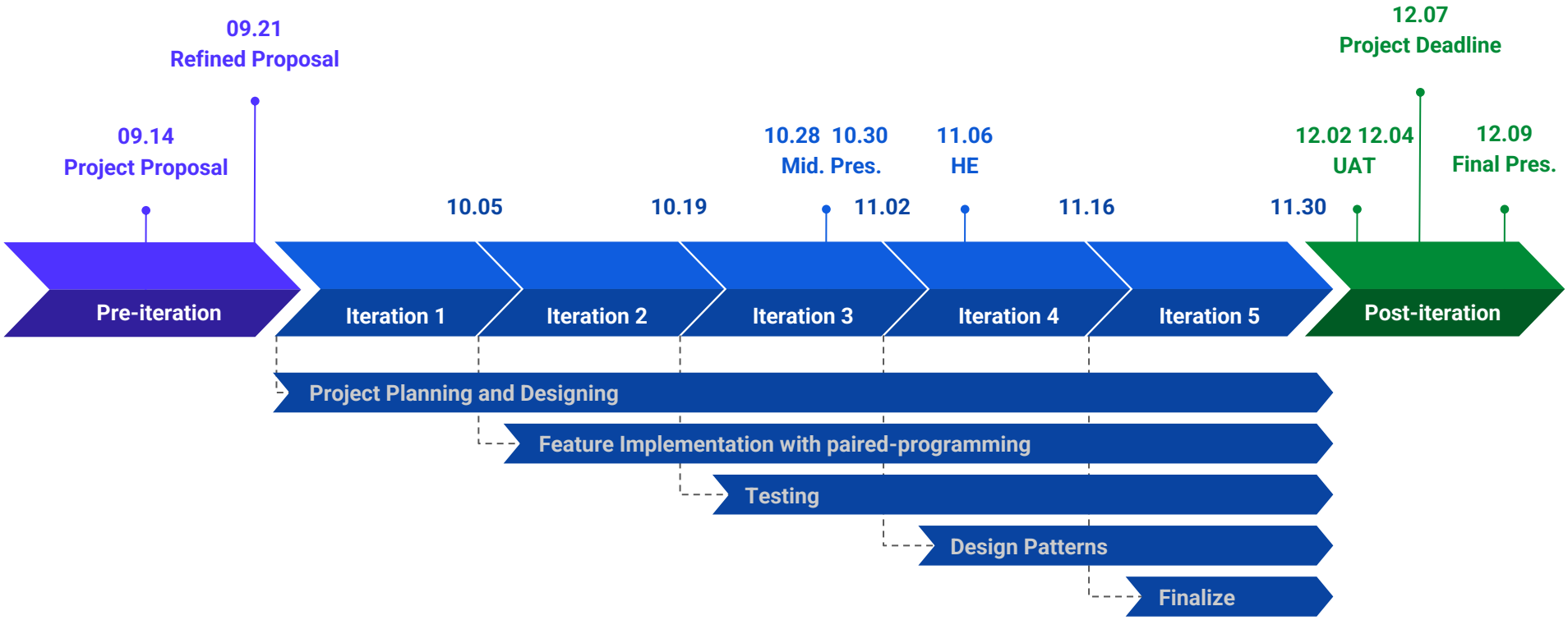
- Lab
  - Thursday 19:00 ~ 20:50
  - Tutorials between Week 1 and Week 6
  - Project meetings and reviews from Week 7 onwards

Tentative Lesson Plan

	Lecture (Tue)	Lecture (Thu)	Lab (Thu)	Project
Week 1 (9/2, 4)	Course Overview	Software Process	Android Basics	Team Assignment
Week 2 (9/9, 11)	Course Project Overview	Project Management I	Django Basics	Proposal
Week 3 (9/16, 18)	Project Management II	Scheduling Exercises	Cloud Computing	Refined Proposal
Week 4 (9/23, 25)	Version Control (GIT)	GIT Exercise	Data Manipulation & ML Inference	**Iteration 1:** Project planning and designing
Week 5 (9/30, 10/2)	Requirements & Specification	Creating User Stories	LangChain Basics	
Week 6 (10/7, 9)	(video) Coding Style/ Code Review	(video) Code Style & Review Practice	(video) App Integration	**Iteration 2:** Feature Implementation
Week 7 (10/14, 16)	Software Testing	Testing Practice	TA Meeting: Design Document and Schedule Reviews	

	Lecture (Tue)	Lecture (Thu)	Lab (Thu)	Project
Week 8 (10/21, 23)	Software Testing	Applying Testing	TA Meeting: Git Usage Reviews	**Iteration 3:** Feature implementation
Week 9 (10/28, 10/30)	Midterm Presentation (Oral)	Midterm Presentation (Oral)	TA Meeting: Overall Project Reviews	
Week 10 (11/4, 6)	Usability	Heuristic Evaluation	TA Meeting: UI Improvements	**Iteration 4:** Feature implementation
Week 11 (11/11, 13)	Design Patterns	Design Patterns	TA Meeting: Design Pattern Reviews	
Week 12 (11/18, 20)	Design Patterns	Design Patterns	TA Meeting: Design Pattern Reviews	**Iteration 5:** Finalize
Week 13 (11/25, 27)	Refactoring	Refactoring Practice	TA Meeting: Refactoring Reviews, UAT Prep Session	
Week 14 (12/2, 4)	User Acceptance Test (UAT) App Market Deployment	UAT	TA Meeting: Overall Project Reviews	Buffer/Wrap-up
Week 15 (12/9)	Final Presentation (Poster)			

Project Overview



Assessment

- Project (70%)
  - Project quality
  - Project process
- Quiz (20%)
  - Total 3 Quizzes
- Android tutorial & labs (10%)
- Bonus & penalty
  - Participation (Slack answers, in-class participation…)
  - Attendance

Course Project (70%)

- Freely build a team of **5 members** [link]
  - Those who are not signed up by 9/7 (Sun) 23:59 will be randomly assigned to a team with fewer than 5 members
- Project topic will be free of choice
- Each team builds an innovative mobile app following a development processes and principles
- We will assess the entire development process, not just the final software
- More details on Week 2!

Quiz (20%)

- Three short quizzes (<30 minutes each)
  - Oct 16. 7PM-9PM
  - Nov 13. 7PM-9PM
  - Dec 04. 7PM-9PM
- Scope: Lecture and Lab materials
- Closed book

Android Tutorials & Labs (10%)

- We will have 5 weeks of Android tutorial lab session
  - Week 1: Android Basics
  - Week 2: Django Basics
  - Week 3: Cloud Computing
  - Week 4: Data Manipulation & ML Inference
  - Week 5: LangChain Basics
  - Week 6: (video) App integration
- Submit your code for exercises in the tutorial
  - More details will be announced in each lab

Attendance

- We will use the electronic attendance system
- You can miss total three classes (including labs) without providing a reason
- 2% (of the final grade) will be taken off for each absence beyond the allowed number of absence
- If you have a valid reason to miss more classes, submit the proof to TAs (via email) within a week of the event
- We do not check for Week 1 and make-up lectures

Attendance

- It is considered as an attendance if you are late for up to 10 minutes
- It is considered as a “*partial*” attendance if you are late for more than 10 minutes
- Being late for more than 30 minutes will be considered as an absence
- Three partial attendance is treated as one absence

Participation

- Delivering interactive and participative learning experience
- You (as a student)
  - Be prepared for class
  - Have a positive impact on collaborative learning
    - Contribute and share what you know
    - Discuss and work in a team
  - Be professional
    - No hijacking the airwaves
    - Give due respect to your peers (listen when they talk)
- We (as the instructor and TAs)
  - Our responsibility and professional duty to assess you

Participation

- We will give small bonus points to those who participate actively in class and on Slack
  - In-class: Ask questions and share your ideas!
  - Slack: Answer each other (e.g., implementation issues, process related matters, etc.)

Course Drop Policies

- Course withdrawal will not be allowed after the end of the official course drop period (09/07)
- The dropout will impact the team project significantly

Course Policies: Academic Integrity

- All acts of academic dishonesty (including, but not limited to, plagiarism, cheating, fabrication, unauthorized possession of exam questions, or tampering with the academic work of other students) are serious offences

Course Policies: Code Plagiarism

- All work submitted for purposes of assessment must be the student's own work
- Penalties for violation of the policy range from zero marks for the component assessment to expulsion, depending on the nature of the offence
- When in doubt, consult the instructor or TAs

Learning References

There are no official textbooks. Recommendations:

- Software Engineering at Google (link)
- Clean Code - Robert C. Martin
- Clean Architecture - Robert C. Martin
- Cornell Uni. Software Engineering 2022 - Prof. Curran Muhlberger (link)
- Seoul Nat. Uni. SWPP2022 - Prof. ByungGon Chun (link)
- Rutgers Uni. Software Enginnering 2020 (link)

Expectations

- ⏰Invest time and self-learn
- 🙏Respect and help your teammates
- 💡Be creative and innovative
- 💪Don’t get frustrated or discouraged
- 💙Be honest

Most importantly…

Let’s have fun and learn from each other

We are here to help!

Notes

- We will start Android tutorials starting this week's lab
- Please setup Android Studio before the lab class
  - Refer to "Tutorial_00_Setup_Android_Studio" slide on eTL

Thank You.
Any Questions?