Programmazione a Oggetti (09CBIxx)

A.A. 2024/2025

Corso A-G

http://softeng.polito.it

Version 1.3.1

Docenti

- Marco Torchiano
  - Dip. Automatica e Informatica
    - IV Piano, Ufficio 4E-33

☎ 011 564 7088

✉ marco.torchiano @ polito.it

🌐 https://softeng.polito.it/torchiano

🐦 @mtorchiano

- Stefano Mancini

✉ stefano.mancini @ polito.it

Modalità di lavoro proposta

Tre tempi

- ↶ Prima delle lezioni
- ▼ Durante le lezioni ufficiali
- ↝ Altri momenti

Modalità di lavoro

- ↶ Prima
  - Video-lezioni asincrone pubblicate in anticipo
- ▼ Durante orario ufficiale
  - Lezioni sincrone
    - Riepilogo contenuti (delle lezioni asincrone)
    - Esempi ed esercizi
    - Domande e chiarimenti
    - Discussione
- ↝ Altri momenti
  - A richiesta: domande, chiarimenti e discussione
  - NON in tempo reale

Strumenti di collaborazione

- Virtual Classroom @ PoliTo
  - Lezioni in streaming + registrazioni
  - Unidirezionale (no chat!)
- Dropbox
  - Cartella del corso con tutto il materiale
    - https://www.dropbox.com/scl/fo/b4id60ykds9xc4hohshi6/ABZZUpHgYLfH024Udz5TAnw?rlkey=io33yyajwygbr8zf0wcotzdqe&st=yldypo18&dl=0
- Telegram
  - Comunicazioni, annunci e interazioni
    - https://t.me/PO_PoliTo

Orario

- Lunedì 10.00 – 11.30
  - Aula 11T
- Mercoledì 16.00 – 19.00
  - Aula 11T
- Giovedì 11.30 – 13.00
  - Aula 11T
- Giovedì 8.30 – 11.30
  - Laib 2B
  - Due squadre a settimane alterne

Laboratori a partire da seconda settimana (6 Marzo)

Calendario Laboratori

	Squadra 1 A – CON	Squadra 2 COR – G
Lab 0 – Git	6/3 (8:30)	6/3 (10:00)
Lab 1 – Basics	13 / 3	20 / 3
Lab 2 – Inheritance	27 / 3	3 / 4
Lab 3 – Collections	10 / 4	17 / 4
Lab 4 – Stream	8 / 5	15 / 5
Lab 5 – I/O	22 / 5	29 / 6
Lab Riepilogo	5 / 6	5 / 6

Inclusività

Una breve premessa

ə

Carta dei Diritti Fondamentali dell’UE

Titolo III – Uguaglianza

Articolo 21 – Non Discriminazione

1. È vietata qualsiasi forma di discriminazione fondata, in particolare, sul **sesso**, la **razza**, il **colore della pelle** o **l'origine etnica o sociale**, le **caratteristiche genetiche**, la **lingua**, la **religione** o le **convinzioni personali**, le **opinioni** politiche o di qualsiasi altra natura, l'appartenenza ad una **minoranza** nazionale, il **patrimonio**, la **nascita**, la **disabilità**, l'**età** o l'**orientamento sessuale**.

https://fra.europa.eu/it/eu-charter/article/21-non-discriminazione

Diritti Fondamentali

- La persona che non li comprende e rispetta è una mezza persona
- L’ingegnere che non li considera è un professionista incompleto
- Ogni linea di codice che scrivete comporta una decisione morale ed etica
- Per mia abitudine e comodità userò spesso il genere grammaticale maschile

Benvenutə
a
tuttə

COURSE ORGANIZATION

Topics

- Software Engineering
  - Software Life Cycle
  - Design
  - Test
  - Configuration management
  - Object-oriented paradigm
- Java programming language
  - Java syntax
  - Standard libraries

Objectives

- Understand how software development works
- Become familiar with the basic development support instruments
- Learn the Java language
- Acquire capability to write and test simple Java programs
- Learn using development tools

Tools

git

Java

JUnit

UNIFIED
MODELING
LANGUAGE

UML™

Organization of the course

- Lectures (~50h)
  - Software Engineering (~15h)
  - Java (~35h)
- Classroom exercises (~20h)
  - Examples (~10h)
  - Assignment solutions (~10h)
- Lab work (~15h)
  - Every week (since W3)

Labs

- LAIBs
  - 1.5h with Teaching + Student Assistants
  - 1.5h with Student Assistant
- Assignments
  - Programs to be completed/modified
  - Similar process as in the final exam
- Assessed and graded
  - Essential for final exam
  - You must be able to use all the software tools in order to pass the exam

The only way to learn a programming language is by coding.

This is the way!

Prerequisites

- Mandatory
  - Procedural programming (e.g. C)
- Recommended
  - Abstract data types
    - Lists, trees etc.
  - Algorithms
    - Sort, search, list insert etc.

Initial self-assessment

- Do you know enough ''C''?

https://moodle.polito.it/mod/quiz/view.php?id=9322

Self-assessment questions

- Proposed usually before labs
- Instrument to enable your self-assessment
  - Useful for us to detect possible problems
- Web based
  - Not anonymous
  - Results not used for grading

Software

- Mandatory
  - JDK 17.0
    https://docs.aws.amazon.com/corretto/latest/corretto-17-ug/downloads-list.html
  - Visual Studio Code + Java Extension Pack
    - https://code.visualstudio.com
- Useful
  - Any UML modeling tool

Evaluation

Evaluation

- Programming two parts (85%)
  - Lab Assignments (30%)
  - Project, two alternatives (55%)
    - Team Project Work
    - Exam Project
- Theory (15%)
  - Closed answer questions

Team Project Work

- Carried on by groups of three students.
- The project must be developed adopting the git–flow approach.
  - In GitLab: using issues, branches, merge requests, reviews
- The project requirements are published in the beginning of May.
- Review meetings with teachers in lab hours
- The fully working project must be delivered by the end last week of lectures (June 6)

Team Project Assessment

- Correctness of implementation
  - acceptance test suite
- Conformance with the recommended process
  - GitLab workflow operations
- Quality of the implementation
  - adherence to good coding practices

Exam Project

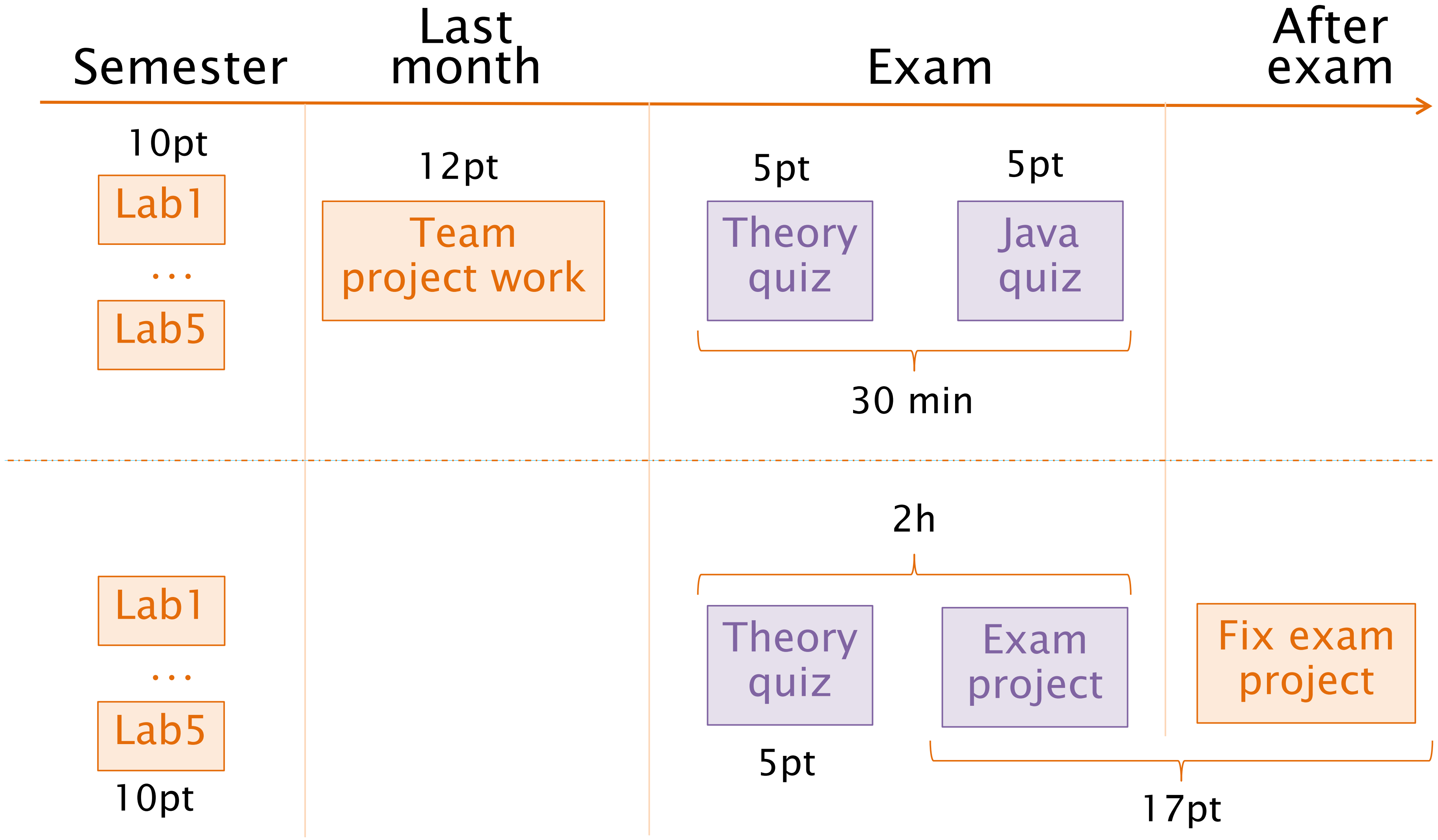
- Phase 1 – in the lab, at exam time
  - Develop Java application, given
    - a textual specification of requirements
    - a skeleton code for the main functions
  - Submit initial version

- Phase 2 – at home, after acceptance tests
  - Check acceptance tests results
  - Fix the app
  - Submit final version
    - Within given deadline (~5 days)

Exam Project Assessment

- Functional correctness
  - Proportion of tests passed by the program version delivered in the lab
- Rework to fix / complete program
  - Number of changes between lab version and final version

Evaluation Summary



Readings

Readings – Java

- Java Documenation
  - https://docs.oracle.com/en/java/javase/17/
- Arnold, Gosling, Holmes. “The Java Programming Language – 4th edition”, Addison-Wesley, 2006
- B.Eckel, “Thinking in Java”, Prentice Hall, 4th Ed., 2006
  - https://www.mindviewllc.com/quicklinks/
- R. Urma, M. Fusco, A. Mycroft. “Modern Java in Action: Lambdas, streams, functional, and reactive programming.” Manning, 2019.
  - https://www.manning.com/books/modern-java-in-action
- B.Eckel. “On Java 8”, Mindview, 2018
  - http://www.onjava8.com/

Readings – Sw Engineering

- Bruegge, Dutoit. *Object-Oriented Software Engineering Using UML, Patterns, and Java*. Pearson, 2009
- *ISO/IEC/IEEE Std 12207-2008 for Systems and Software Engineering – Software Life Cycle Processes*
    - http://ieeexplore.ieee.org/document/4475826/

Readings – Test

- ISO/IEC/IEEE, Std 29119-1 Software and systems engineering – Software testing – Part 1: Concepts and definitions, 2013.
- ISTQB, Certified Tester Foundation Level Syllabus, 2001
  - http://www.istqb.org/downloads/send/2-foundation-level-documents/3-foundation-level-syllabus-2011.html4

Readings – Config Management

- Collins-Sussman, Fitzpatrick, Pilato. *Version Control with Subversion*, 2001
    - http://svnbook.red-bean.com
- IEEE Std 828-2012 *Standard for Configuration Management in Systems and Software Engineering*, 2012
- Semantic Versioning
    - http://semver.org

Readings – Design

- M.Fowler, K. Scott, *UML Distilled*, 3rd ed. Addison-Wesley, 2003.
- E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA: Addison-Wesley, 1995.
- E.Freeman, E.Freeman, K.Sierra, B.Bates. *Head First Design Patterns*, O'Reilly, 2004