Programmazione a Oggetti
(09CBIxx)

A.A. 2020/2021

Corso A-DIM

Docenti

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

☎ 011 564 7088
✉ marco.torchiano @ polito.it
🌐 http://softeng.polito.it/torchiano
@mtorchiano

- Diego Monti

✉ diego.monti @ polito.it

- Simone Leonardi

✉ simone.leonardi @ polito.it

Modalità di lavoro proposta

Tre tempi

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

Modalità di lavoro proposta

- ↶ Prima
  - Video-lezioni asincrone pubblicate in anticipo
  - Con notifica
- ▼ Durante orario ufficiale
  - Lezioni sincrone con interazione audio-video
    - Riepilogo contenuti (presentati in lezioni asincrone)
    - Esempi
    - Domande e chiarimenti
    - Discussione
- ⇝ Altri momenti
  - A richiesta: domande, chiarimenti e discussione
  - NON in tempo reale

Strumenti di collaborazione

- Virtual Classroom @ PoliTo
  - Lezioni
  - Laboratori (supporto online)
    - http://www.politocomunica.polito.it/press_room/didattica_on_line

slack

  - Supporto e interazioni “continue” su lab
  - Con possibili interazioni individuali
  - https://po2021polito.slack.com/signup
    - Registrarsi con email PoliTo

Pagina del corso

https://softeng.polito.it/courses/09CBI

- Notizie ed informazioni
- Materiale
  - Slides,
  - Esercizi
  - Strumenti
  - Videolezioni
    - Anche su playlist YouYube: http://bit.ly/37UxucM

Regole e informazioni

https://oop.polito.it/doc/

- Software di Riferimento
- Modalità d'Esame
- Avvio delle Esercitazioni di Laboratorioaa
- Uso dei progetti SVN in Eclipse
- Installazione Subversion Plug-in
- Uso dei test nei file Jar

Orario

- Lunedì 8.30 – 11.30
  - Aula Virtuale
- Giovedì 10.00 – 13.00
  - Aula 5 (forse)

- Mercoledì 16.00 – 19.00
  - Aula Virtuale

Laboratori a partire dalla terza settimana (17 Marzo)

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
- Know the Java language
- Write and test simple Java programs
- Learn using development tools

Tools

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 but not graded
  - Essential for final exam
  - You must be able to use all the software tools in order to pass the exam

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''?
- Should you review it a bit?

http://bit.ly/3bpjYiy

- Or http://softeng.polito.it/survey/index.php/464583
- Use your id (matricola)
  - Numbers only, no initial «s»

Self-assessment questions

- Proposed during the course
- A set of closed answer questions
- 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 11.0
    https://docs.aws.amazon.com/corretto/latest/corretto-11-ug/downloads-list.html
  - Eclipse IDE – 2020-12
    - https://www.eclipse.org/downloads/packages/
  - Subclipse plug-in for Eclipse
    - Installed from within Eclipse: *Help > Eclipse Marketplace*
- Useful
  - Astah UML – (free student license)
    - http://astah.net/editions
  - Papyrus plug-in for Eclipse

https://oop.polito.it/doc/ReferenceSoftware_it.html

FINAL EXAM

Final Exam – Programming

- Abilities verified
  - Analyze simple requirements
  - Design a solution to address problem
  - Write correct and complete Java program
  - Use development tools
  - Understand tests and their reports

Final Exam

- Part I: Programming (~85%)
  - Step I: during exam write the code
  - Step II: at home fix the code
- Part II: Theory (~15%)
  - Closed answer written questions
- Rules
  - 2 hours

Final exam – Programming

- Phase 1 – on your PC, 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, later
  - Receive acceptance tests
  - Fix the app
  - Submit final version
    - Within given deadline (~5 days)

Specific technical details not defined yet

Final Exam – Assessment

- Programming
  - Functional correctness
    - Proportion of tests passed by the program version delivered in the lab
  - Rework to fix / complete program
    - Amount of changes between lab version and final version
- Theory
  - Correct answers

Readings – Java

- Java Documenation
  - http://www.oracle.com/technetwork/java/javase/documentation/index.html
- 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