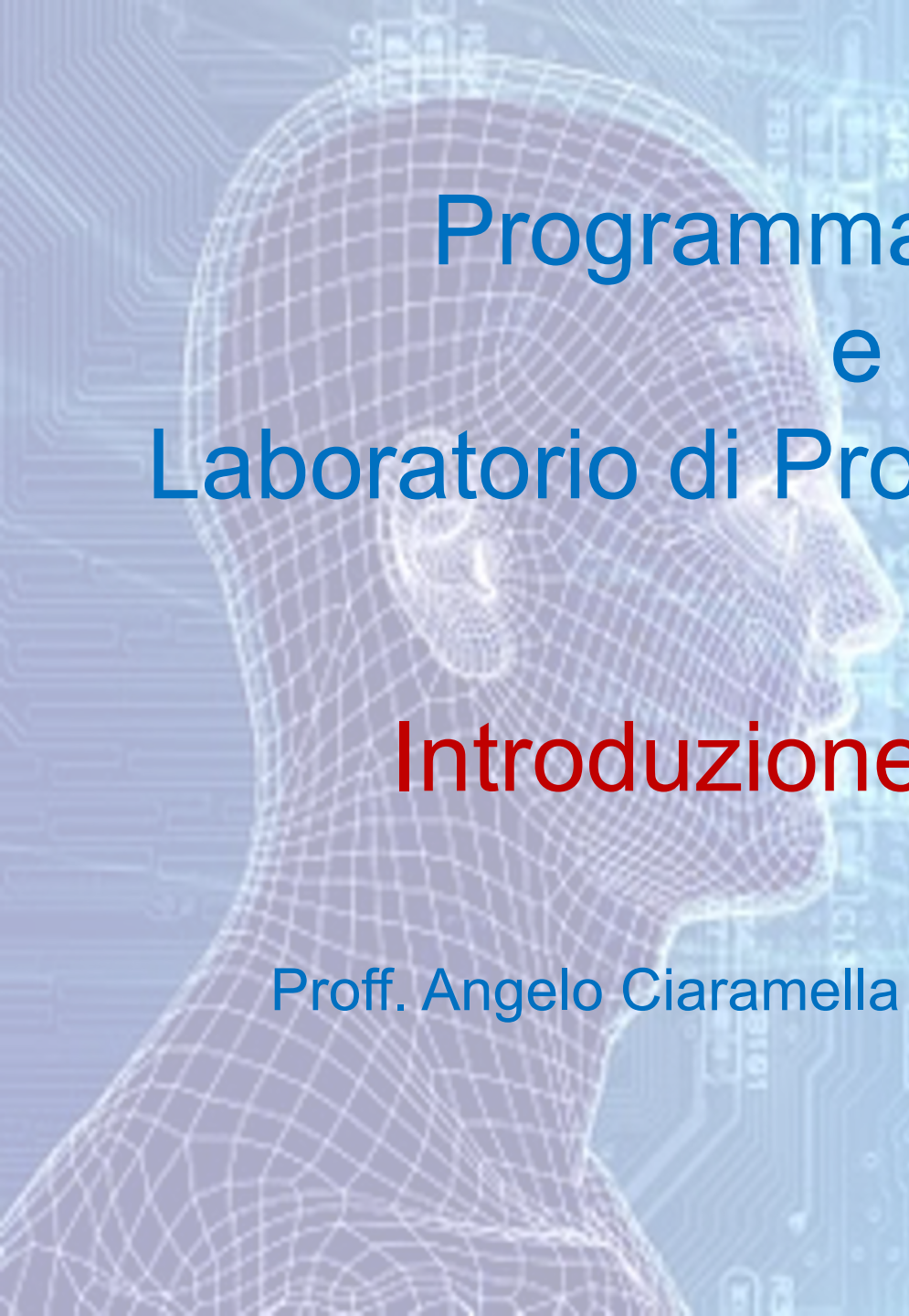
UNIVERSITÀ PARTHENOPE
DiST
Dipartimento di Scienze e Tecnologie



Programmazione 3
e
Laboratorio di Programmazione 3

Introduzione al corso

Proff. Angelo Ciaramella – Emanuel Di Nardo

Crediti ed esami

- Corso formato da 6 CFU (48 ore)
  - 3 CFU – Prof. Angelo Ciaramella (Parte I)
  - 3 CFU – Prof. Emanuel Di Nardo (Parte II)
- Lezioni
  - Frontali
  - Esercitazioni
    - Esercitazioni teoriche
    - Esercitazioni di laboratorio
- Modalità di esame
  - Progetto pratico
  - Colloquio orale
    - Presentazione

Orario lezioni e ricevimento

- Orario lezioni
  - Martedì
    - 9:00 - 11:00 – Aula 18
  - Giovedì
    - 14:00 - 16:00 – Aula 2
- Codice Teams
  - **p3vpsq3**
- Orario di ricevimento
  - Piattaforma docenti
  - "On demand"
    - via e-mail

Obiettivi del corso

- Il corso intende fornire concetti avanzati per la programmazione orientata agli oggetti

- Il corso sarà suddiviso in due parti principali
  - Programmazione orientata agli oggetti
    - Object-Oriented Programming (OOP)
    - Esercitazioni pratiche in Linguaggio Java
  - Metodologie di analisi e progettazione orientate agli oggetti
    - Object Oriented Analysis (OOA)
    - Object Oriented Design (OOD)

Materiale didattico

- Testi consigliati
  - OOP
    - *Il nuovo Java. Guida completa alla programmazione moderna,* C. De Sio Cesari, Hoepli, 2020
  - OOA/OOD
    - *Design Patterns: Elements of Reusable Object-Oriented Software*, E. Gamma, R. Helm, R. Johnson, J. Vlissides (the GangOfFour), AddisonWesley Professional, 1994
    - *Design Pattern in Java (manuale pratico)*, S. J. Metsker, Pearson Addison Wesley, 2003
    - *Clean Code*, R. C. Martin, Prentice Hall, 2009
- Download
  - Materiale docente sulla piattaforma e-learning

Programma del corso

- OOP
  - In questa parte del corso saranno affrontati i concetti avanzati della programmazione ad oggetti comuni a tutti i linguaggi
  - Partendo da un approfondimento teorico dei costrutti di base si passerà all'analisi dei concetti costituenti il paradigma OO fino ai principi avanzati di astrazione
  - Il corso prevede una parte di esercitazioni pratiche sui concetti appresi, fatta in laboratorio in linguaggio Java

Programma del corso

- Schema riassuntivo
  - Concetti di base
    - Classe, metodi, costruttori, *overloading*, *overriding*, vettori, array, flussi, …
  - Principi di *information hiding* ed incapsulamento
  - Ereditarietà
  - Polimorfismo (dynamic binding)
  - Strategie di astrazione
    - classi astratte ed interfacce (interface)
  - Gestione delle eccezioni
  - Materiale per i progetti
    - interfacce utente grafiche
    - Gestione di eventi
    - Java Database Connectivity
    - JShell
  - Esercitazioni pratiche in linguaggio Java

Programma del corso

- OOA/OOD
  - Questa parte del corso riguarderà i principi base di analisi e design del paradigma OO
    - con particolare riferimento ai design pattern architetturali classici, definiti in letteratura (23 di GoF)
  - Partendo da tecniche di base di progettazione e design saranno man mano approfonditi i principi che guidano le scelte architetturali avanzate, applicabili al paradigma OO

Programma in dettaglio

- Schema riassuntivo
  - Identificazione dei requisiti
    - Concetti, attività, gestione
  - OOA
    - Analisi dei concetti, delle attività, della gestione
    - Approccio Class-Responsibility-Collaborator
  - OOD
    - System Design
    - Object Design
  - Principi di progettazione (Agile design)
    - Single Responsibility
    - Dependency Inversion
    - Legge di Demeter
    - Open/Closed
    - Sostituzione di Liskov
    - Interface Segregation

Programma del corso

- Schema riassuntivo
  - Elementi di UML (Unified Modeling Language)
  - La programmazione secondo i pattern (GoF)
    - Creazionali (Creational Design Patterns)
    - Strutturali (Structural Design Patterns)
    - Comportamentali (Behavioral Design Patterns)

OOA → OOD → OOP

un concetto del dominio

OOA

Aereo
codiceDiRegistrazione

visualizzazione di un concetto del dominio

OOD

Aereo
- codiceDiRegistrazione
+ getVoliEffettuati: List

una classe software

OOP

```
public class Aereo {
    private String codiceDiRegistrazione;

    public List getVoliEffettuati() {...}
}
```

rappresentazione in un linguaggio di programmazione orientato agli oggetti