POLO SCIENTIFICO TECNICO PROFESSIONALE
E. FERMI - G. GIORGI

Programma Didattico 2023/2024

Prof. Alessandro Baroni 4 Informatica

Di seguito si farà riferimento al libro con la sintassi “Unità Ax – Ly” indicando l’unità didattica x e la lezione y ripresa dall’indice del testo utilizzato.

Testo di Riferimento
Corso di Informatica. 2E Volume 2 - Fiorenzo Formichi, Giorgio Meini, Ivan Venuti – Zanichelli
Materiale didattico aggiuntivo
Pubblicato sulla piattaforma didattica “Classroom” della suite di Google.

Unità **D**idattica di **A**pprendimento: Introduzione alla programmazione ed alla progettazione orientata agli oggetti

Conoscenze

- Conoscere la modellazione ad oggetti, il concetto di information hiding, ed alcuni strumenti di modellazione UML (diagramma delle classi), le associazioni fra le classi ed il concetto di eredità e polimorfismo.
- Espressioni, istruzioni, cicli, funzioni, classi, oggetti, ereditarietà, incapsulamento, polimorfismo.
- Principi della programmazione orientata agli oggetti (OOP) e loro applicazioni pratiche.
- Linguaggi di programmazione Java per l'OOP.
- Strutture dati fondamentali: array, liste, pile, code, alberi, grafi.
- Concetti di progettazione del software: modularità, design pattern.
- Concetti di algoritmi.

Abilità

- Nel modellare un tipo di dato astratto, ed eventuali relazioni fra le varie istanze di oggetti.
- Nel tradurre problemi reali in algoritmi e codice eseguibile.
- Nell'analizzare, comprendere e modificare codice sorgente esistente.
- Nell'organizzare e gestire progetti software complessi.
- Nel collaborare e comunicare efficacemente all'interno di un team di sviluppo software.
- Nel risolvere errori e debuggare codice.
- Nell'ottimizzazione delle prestazioni del software.

Competenze

- Capacità di scrivere codice sorgente utilizzando un linguaggio di programmazione orientato agli oggetti.
- Abilità nella risoluzione di problemi attraverso l'applicazione di algoritmi e strutture dati.
- Competenza nell'uso di IDE (Integrated Development Environment) e strumenti di sviluppo per la scrittura, il debug e il testing del software.
- Capacità di progettare e implementare classi e oggetti per risolvere problemi specifici.
- Abilità nell'utilizzo di design pattern per la creazione di software robusto e manutenibile.
- Capacità di comprendere e utilizzare librerie e framework orientati agli oggetti.

Contenuti:

1. Tipi di dato astratto e principio di Information hiding (Unità A1 – L1).
2. Classi ed oggetti, attributi e metodi nei diagrammi UML (Unità A1 – L2).
3. Interazione tra gli oggetti e diagrammi UML di sequenza (Unità A1 – L3).
4. Ereditarietà e polimorfismo (Unità A1 – L4).
5. Associazioni tra le classi (Unità A1 – L5).

Laboratorio

Esercitazioni su Visual Paradigm Online per modellazione classi, ed introduzione all'ambiente di sviluppo Eclipse.

Unità **D**idattica di **A**pprendimento: Il linguaggio di programmazione Java.

Conoscenze

- Conoscere il linguaggio ibrido Java ed il concetto di byte-code.
- Conoscere la differenza del pacchetto JDK e JRE con relativo funzionamento della Java Virtual Machine.
- Sintassi del Linguaggio
- Comprendere la sintassi di base del linguaggio Java, inclusi tipi di dati, operatori, strutture di controllo, etc.
- Utilizzare le funzionalità avanzate del linguaggio, come le espressioni lambda (nel gestire gli eventi del pattern Model View Controller) e le interfacce funzionali.
- Gestione della Memoria e Garbage Collection
- Comprendere il funzionamento della gestione della memoria in Java.
- Utilizzo delle API Standard
- Conoscere le API standard di Java per l'accesso al filesystem, la manipolazione delle stringhe, la gestione delle date, etc.
- Utilizzare le API standard per sviluppare applicazioni Java robuste e affidabili.

Abilità

- Progettazione e implementazione di Classi
- Progettare classi e definire le loro relazioni in base ai requisiti del problema.
- Implementare le classi utilizzando i concetti di incapsulamento e coerenza.
- Risoluzione dei problemi:
- Utilizzare il linguaggio Java per risolvere problemi di programmazione complessi.
- Sviluppare soluzioni efficienti ed efficaci utilizzando le caratteristiche del linguaggio.
- Debugging e Testing: utilizzare strumenti di debugging per identificare e risolvere errori nel codice Java.
- Scrivere test unitari per verificare il corretto funzionamento delle componenti software.
- Collaborazione e comunicazione
    - Lavorare in gruppo per sviluppare progetti Java collaborativi.
    - Comunicare in modo chiaro e efficace le soluzioni proposte e le sfide incontrate durante lo sviluppo.

Competenze

- Identificare e spiegare i concetti chiave del paradigma orientato agli oggetti, come incapsulamento, ereditarietà, e polimorfismo.
- Descrivere la relazione tra classi e oggetti nel contesto di Java.
- Gestione delle Classi e degli Oggetti
- Creazione classi in Java con attributi e metodi appropriati.
- Istanziare oggetti dalle classi definite.
- Implementare costruttori per inizializzare gli oggetti.
- Ereditarietà e Polimorfismo:
    - applicare il concetto di ereditarietà per estendere le funzionalità delle classi esistenti.
    - Utilizzare il polimorfismo per scrivere codice più flessibile e riutilizzabile.
- Gestione delle Eccezioni
    - Identificare e gestire le eccezioni che possono verificarsi durante l'esecuzione di un programma Java.Implementare la gestione delle eccezioni per garantire la robustezza del codice.

- Utilizzo delle Collezioni
    - Comprendere le diverse strutture dati fornite dalle collezioni Java, come ArrayList, LinkedList, HashMap, etc.
    - Utilizzare le collezioni per organizzare e manipolare dati in maniera efficiente.
- Gestione input/output
- Leggere e scrivere da file, utilizzare lo standard Comma Separated Value (CSV)
- Serializzazione degli oggetti.

Laboratorio

Implementazione su Eclipse di algoritmi.

Unità **D**idattica di **A**pprendimento: Tipi generici e collezioni nel linguaggio Java.

Conoscenze

- Concetti fondamentali dei tipi generici nel linguaggio Java: definizione, utilizzo, vantaggi.
- Tipi di collezioni disponibili nel framework delle collezioni di Java: List, Set, Map, Queue, Stack, etc.
- Implementazioni specifiche delle collezioni: ArrayList, LinkedList, HashMap etc.
- Concetti di iterazione e manipolazione delle collezioni.
- Principali operazioni e metodi offerti dalle collezioni nel linguaggio Java: aggiunta, rimozione, ricerca, ordinamento, etc.
- Utilizzo delle classi wrapper e autoboxing per gestire i tipi primitivi all'interno delle collezioni.
- Programmazione generica

Abilità

- Abilità nell'applicare i concetti di tipi generici e collezioni per risolvere problemi pratici di programmazione.
- Abilità nell'utilizzare i metodi e le operazioni offerti dalle collezioni Java per manipolare e gestire i dati in modo efficiente.
- Capacità di scrivere codice chiaro, leggibile e manutenibile che fa uso delle collezioni e dei tipi generici.
- Abilità nel valutare le prestazioni delle diverse implementazioni delle collezioni e selezionare quella più adatta alle esigenze del progetto.
- Abilità nel progettare e implementare algoritmi che utilizzano tipi generici e collezioni per risolvere problemi complessi.

- Abilità nell’implementare classi che gestiscono tipi generici.

Competenze

- Capacità di definire e utilizzare tipi generici per creare codice flessibile e riutilizzabile.
- Abilità nell'utilizzare le diverse implementazioni delle collezioni Java in base alle esigenze specifiche del problema.
- Competenza nell'iterare e manipolare gli elementi delle collezioni utilizzando iteratori, forEach loop, etc.
- Capacità di gestire eccezioni e problemi legati alle operazioni sulle collezioni, come gestione di elementi duplicati, elementi nulli, etc.
- Abilità nella scelta delle strutture dati più appropriate per ottimizzare le prestazioni del software.

Laboratorio

Implementazione su Eclipse di strutture dati, tipi generici e progetti di gruppo.

Unità **D**idattica di **A**pprendimento: Graphical User Interface con JavaFX

Conoscenze

- Concetti fondamentali delle interfacce grafiche utente (GUI) e dei principi di progettazione dell'interfaccia utente, come il pattern di modellazione Model View Controller (MVC).
- Architettura e struttura di JavaFX come framework per la creazione di interfacce utente.
- Componenti di base di JavaFX: Scene, Stage, Layouts, Controls, Shapes, e l'organizzazione gerarchica della scena.
- Gestione degli eventi nell'interfaccia utente: event handling, listener, e gestione degli eventi di input utente.
- Utilizzo di CSS per lo stile e la formattazione delle interfacce utente JavaFX.
- Integrare elementi multimediali e grafici all'interno dell'interfaccia utente JavaFX.

Abilità

- Nel tradurre requisiti funzionali e di progettazione in un'interfaccia utente JavaFX.
- Nel risolvere problemi di progettazione e implementazione legati all'interfaccia utente.
- Di testare e debuggare le interfacce utente JavaFX per garantire la funzionalità e l'usabilità.
- Nel documentare e presentare il processo di progettazione e sviluppo dell'interfaccia utente.
- Capacità di collaborare efficacemente all'interno di un team per lo sviluppo di applicazioni JavaFX.

Competenze

- Capacità di progettare e sviluppare interfacce utente intuitive e funzionali utilizzando JavaFX.
- Abilità nel creare layout flessibili e adattabili che si adattano a diverse dimensioni e risoluzioni dello schermo.
- Competenza nell'implementare e gestire eventi utente come click del mouse, pressione dei tasti, movimenti del mouse, etc.
- Capacità di utilizzare CSS per personalizzare l'aspetto e lo stile degli elementi dell'interfaccia utente.
- Abilità nella gestione e manipolazione dei dati all'interno dell'interfaccia utente attraverso i controlli JavaFX.

Laboratorio

Implementazione su Eclipse di piccoli gestionali con interfaccia grafica, progetti a gruppo per sviluppare le capacità di interazione, dialogo e collaborazione.