

# Laboratorio di strumenti avanzati a supporto dello sviluppo software

## Contenuti

Il corso si propone di illustrare come strumenti e metodologie di software configuration management possano essere utilizzate a supporto di progetti software in ambito educativo, al fine di stimolare il lavoro di gruppo e la revisione critica degli artefatti sviluppati.

## Finalità

Le finalità principali del corso sono:

- Comprendere il ruolo del processo di software configuration management, e degli strumenti di cooperative software development
- Comprendere nel dettaglio il funzionamento dei sistemi di versioning, anche nell'ambito di forge di progetti open source
- Comprendere i meccanismi di code review e pull request development
- Utilizzare strumenti a supporto del cooperative work
- Comprendere i principi del continuous integration e delivery

## Indice

- Introduzione al cooperative software development
- Software configuration management
- Sistemi di versioning – cenni storici e funzionamento di git
- Code review, pull based software development
- Uso di webhook con strumenti di comunicazione in progetti software
- Cenni al continuous integration e delivery. GitHub actions

## Lecture consigliate

1. Appunti delle lezioni.
2. Chacon, S., & Straub, B. (2014). *Pro Git* (2nd ed.). Apress, <https://git-scm.com/book/en/v2>

# Fondamenti di programmazione

## Contenuti

Il corso intende fornire ai partecipanti le basi teoriche e pratiche per l'insegnamento della programmazione nella scuola, con particolare attenzione al pensiero computazionale, all'interdisciplinarietà e alla progettazione di attività didattiche efficaci. I partecipanti svilupperanno una comprensione critica dei linguaggi di programmazione per la didattica, dei modelli cognitivi coinvolti nell'apprendimento della programmazione e delle strategie pedagogiche per introdurre algoritmi e strutture dati in contesti scolastici.

## Finalità

1. Comprendere e applicare il pensiero computazionale
2. Integrare il pensiero computazionale nelle discipline scolastiche
3. Conoscere e utilizzare linguaggi di programmazione didattici
4. Comprendere i modelli mentali nella didattica della programmazione
5. Progettare attività didattiche efficaci

## Indice

- Pensiero computazionale e didattica trasversale
- Applicazioni del pensiero computazionale in diverse discipline (arte, letteratura, fisica)
- Didattica della programmazione
- Notional Machine e Modelli Mentali nella Didattica della Programmazione
- Linguaggi di Programmazione per la didattica
- Didattica degli algoritmi e delle strutture dati

## Letture consigliate

1. Maureen D. Neumann, Lisa Dion, Robert Snapp  
*Insegnare il pensiero computazionale. Un approccio integrato per l'apprendimento nella scuola secondaria di primo e secondo grado*  
Roma: Edizioni Anicia, 2024
2. Jeannette Wing, *Computational Thinking*, Communications of the ACM, 2006.
3. Orit Hazzan, Noa Ragonis, Tami Lapidot  
*Guide to Teaching Computer Science: An Activity-Based Approach*  
Springer, 2020 (3ª edizione)

# **Fondamenti di Didattica nell'Informazione**

## **Contenuti**

Il corso è incentrato sulle tematiche principali riguardanti la didattica dell'informatica.

## **Finalità**

La finalità principale del corso è porre i corsisti nella condizione di progettare e attuare interventi formativi sui principali argomenti riguardanti l'informatica nelle scuole secondarie superiori

## **Indice**

- Introduzione, struttura e obiettivi del corso
- La didattica dell'informatica
- La didattica dei linguaggi di programmazione
- Cenni sui paradigmi di apprendimento
- Approccio top-down e bottom-up
- Imparare facendo
- Progettazione didattica
- Didattica della programmazione
- La valutazione

## **Letture consigliate**

1. Appunti e dispense del docente

# Laboratorio di Tecnologie per lo Sviluppo di Sistemi Software

## Contenuti

Presentazione del contesto didattico sullo sviluppo software: ciclo di vita del software, ambienti di versioning e IDE; analisi dei requisiti e progettazione di un sistema software, UML e design patterns; linguaggi di programmazione e framework; basi di dati e SQL; applicazioni web; approcci didattici con laboratori, progettazione di unità di apprendimento e ambienti educativi.

## Finalità

Fornire ai partecipanti conoscenze operative e didattiche sui principali strumenti e le principali metodologie per lo sviluppo software in un contesto scolastico-laboratoriale. In particolare, si prevede di: consolidare competenze tecniche e metodologiche per la progettazione, lo sviluppo ed il testing di progetto software; sviluppare le capacità di selezione ed applicazione di strumenti e tecnologie per lo sviluppo software; promozione di un approccio didattico laboratoriale di tipo collaborativo.

## Indice

- **Introduzione al contesto didattico riguardante lo sviluppo software:** Cenni al ciclo di vita del software; Strumenti di versioning del codice; Ambienti di sviluppo integrati e loro installazione.
- **Progettazione del software:** Cenni ad Analisi dei requisiti funzionali e non funzionali; Diagrammi UML di base e Design Patterns; Introduzione e applicazioni didattiche.
- **Linguaggi di programmazione e ambienti di sviluppo:** Introduzione ai linguaggi di programmazione; Sviluppo di applicazioni; Framework di sviluppo.
- **Basi di dati e interazione con il software:** Modello relazionale (modello E-R); SQL di base e interazione con applicazioni software.
- **Sviluppo web e mobile:** Struttura di una web app
- **Test, debug e deployment:** Tecniche di testing (unit test, integration test); Debugging e gestione degli errori. Principi di CI/CD – Continuous Integration/Deployment.
- **Didattica laboratoriale e valutazione:** Metodologie didattiche per l'insegnamento del coding e dello sviluppo software. Progettazione di unità di apprendimento. Uso ed installazione di ambienti didattici.

## Letture consigliate

1. Dispense, risorse online, documentazione ufficiale degli strumenti utilizzati

# Metodologie per la didattica dell'informazione

## Contenuti

Il corso si concentra sull'insegnamento dei fondamenti teorici, metodologici e operativi per progettare, realizzare e valutare percorsi didattici efficaci in ambito informatico. Particolare attenzione è riservata alla didattica attiva, alla rappresentazione concettuale dei contenuti e alla valutazione dell'apprendimento, con un'integrazione tra informatica, pedagogia e tecnologie didattiche.

## Finalità

Le finalità principali del corso:

- Comprendere le cornici teoriche della didattica dell'informatica.
- Saper analizzare e strutturare contenuti concettualmente complessi.
- Progettare percorsi formativi fondati su approcci attivi e cooperativi.
- Scegliere e utilizzare strumenti didattici digitali e ambienti di apprendimento.
- Valutare efficacemente l'apprendimento in contesti di coding e progettazione algoritmica.
- Adottare strategie inclusive per una didattica attenta alla diversità cognitiva.

## Indice

- Cornici teoriche della didattica dell'informatica
- Approcci pedagogici (comportamentismo, costruttivismo, costruzionismo, connettivismo)
- Tassonomie educative (Bloom, SOLO) e framework per le competenze digitali (DigCompEdu)
- Analisi del dominio concettuale dell'informatica
- Strumenti per la mappatura concettuale (mappe, graph knowledge)
- Progettazione di percorsi didattici: obiettivi, attività, risultati attesi
- Didattica attiva: coding, debugging, attività unplugged, pair programming
- Ambienti digitali per l'insegnamento del coding (Snap!, Thonny, Replit, Jupyter)
- Strumenti per la gestione e il monitoraggio dell'apprendimento (LMS, CodeRunner, Moodle Quiz)
- Metacognizione e pensiero computazionale (astrazione, decomposizione, generalizzazione, valutazione)
- Valutazione formativa e sommativa in ambito informatico
- Costruzione di rubriche per attività di programmazione
- Strategie inclusive (UDL, attenzione a DSA, ADHD, gifted)
- Prospettive di ricerca nella didattica dell'informatica
- Progetti e iniziative internazionali (Bebras, CSTA, Informatics Europe)

## **Lecture consigliate**

1. Appunti delle lezioni.
2. ACM Curriculum Guidelines for Computer Science Education.
3. Informatics Europe – Informatics for All.
4. Bruner, J. – La cultura dell'educazione
5. Papert, S. – Mindstorms: Children, Computers, and Powerful Ideas.
6. Wing, J. – Computational Thinking.

# **Laboratorio di strumenti di sviluppo software avanzati**

## **Contenuti**

Il corso verte sull'insegnamento dei sistemi operativi con particolare riferimento alle conoscenze necessarie per lo sviluppo di sistemi software avanzati, quali processi e file system. Il corso mira a fornire conoscenze pratiche tramite approfondimenti e laboratori con il sistema operativo Linux.

## **Finalità**

Le finalità principali del corso sono:

- Conoscere i sistemi operativi;
- Ampliare le conoscenze necessarie per lo sviluppo di software avanzati;
- Approfondire l'interfaccia utente-sistema operativo;
- Conoscere i fondamenti della programmazione di shell.

## **Indice**

- Introduzione e definizione di sistema operativo;
- Richiami sulle periferiche hardware e sistema di I/O;
- Introduzione al sistema Linux;
- La shell;
- Funzionalità della shell;
- Sviluppo software e shell scripting;
- Variabili, ricerca, lettura e manipolazione dei file;
- Gestione dei processi;
- File system e memoria di massa;
- Sicurezza e gestione dei permessi;
- Approfondimenti e laboratori con il sistema Linux.

## **Lecture consigliate**

1. Appunti delle lezioni.
2. Materiale esercitativo, di approfondimento e risorse disponibili in rete suggerite dal docente.