



Istituto di Istruzione Superiore "Giorgi – Woolf"

Viale Palmiro Togliatti, 1161 – 00155 Roma

Curricolo verticale della disciplina **TECNOLOGIE E PROGETTAZIONE DI SISTEMI INFORMATICI E DI TELECOMUNICAZIONI**

Percorso di studio: **Istituto Tecnico** Indirizzo di studio: **Informatica e Telecomunicazioni**

SECONDO BIENNIO

QUARTO ANNO

MODULO	PREREQUISITI	CONOSCENZE	ABILITÀ	COMPETENZE	ATTIVITÀ DI LABORATORIO	CONNESSIONI INTERDISCIPLINARI	TEMPI
MODULO 1 Fasi e modelli di gestione di un ciclo di sviluppo	•	<ul style="list-style-type: none">• Comprendere la necessità di una metodologia per lo sviluppo di sistemi informatici• Conoscere gli elementi fondamentali dell'ingegneria del software• Conoscere gli elementi teorici della progettazione a oggetti (OOP)• Capire l'utilizzo delle schede CRC per l'identificazione di classi• Conoscere una metodologia di documentazione (UML)• Conoscere le tecniche di schedulazione delle fasi	<ul style="list-style-type: none">• Scegliere le metodologie e le tecniche adeguate alle diverse situazioni• Applicare il concetto di astrazione per modellare le classi• Utilizzare ArgoUML per documentare un progetto• Utilizzare Excel per realizzare diagrammi di Gantt	<ul style="list-style-type: none">• Individuare e descrivere il problema complesso• Usare la progettazione orientata agli oggetti per programmi complessi• Rappresentare classi e oggetti mediante diagrammi UML• Usare i diagrammi UML per descrivere le relazioni tra gli elementi di un progetto• Usare la progettazione orientata agli oggetti per sistemi informatici complessi• Schedulare le fasi di un progetto con i diagrammi di Gantt	<ul style="list-style-type: none">• Realizzare un diagramma di Gantt con Excel• I diagrammi UML con software		Settembre-Ottobre



Istituto di Istruzione Superiore "Giorgi – Woolf"

Viale Palmiro Togliatti, 1161 – 00155 Roma

<p>MODULO 2 I REQUISITI SOFTWARE</p>		<ul style="list-style-type: none"> • Comprendere l'importanza della fase di analisi • Avere il concetto di requisito utente e di sistema • Avere il concetto di fase di esplorazione • Conoscere le tecniche di esplorazione • Avere il concetto di scenario e caso d'uso • Comprendere i principi S.O.L.I.D. per la definizione delle classi 	<ul style="list-style-type: none"> • Saper descrivere in UML i casi d'uso • Saper descrivere in UML il diagramma di contesto • Saper documentare i casi d'uso 	<ul style="list-style-type: none"> • Individuare i requisiti utente e di sistema • Utilizzare le tecniche di esplorazione • Individuare gli scenari d'uso 	<ul style="list-style-type: none"> • Uso dei diagrammi di stato , attività (progettazione algoritmo) in UML e tramite Software applicativo. 	<ul style="list-style-type: none"> • 	<p>Novembre-Dicembre</p>
<p>MODULO 3 DOCUMENTAZIONE DEL SOFTWARE</p>		<ul style="list-style-type: none"> • Comprendere la necessità di documentare • Sapere quali sono i documenti necessari in un progetto • Conoscere il concetto di documentazione interna ed esterna • Apprendere le modalità per realizzarla documentazione esterna di sistema e utente • Acquisire una tecnica di documentazione del codice • Conoscere i principali tool di documentazione automatica del codice 	<ul style="list-style-type: none"> • Saper usare i principali tool di documentazione automatica del codice • Saper definire uno standard di documentazione 	<ul style="list-style-type: none"> • Saper organizzare la documentazione del progetto • Saper formattare il codice • Saper effettuare la documentazione del codice • Saper condividere componenti di un progetto con il team di sviluppo 	<ul style="list-style-type: none"> • Utilizzare Javadoc come strumento di documentazione automatica 	<ul style="list-style-type: none"> • Informatica: Conoscenza della Classi e loro principale Uso. Analisi, studio dei dati principali e stesura di algoritmi e UML 	<p>Gennaio-Febrero</p>
<p>MODULO 4 PROCESSI SEQUENZIALI E PARALLELI</p>		<ul style="list-style-type: none"> • Conoscere i modelli di elaborazione dei processi • Conoscere il ciclo di vita dei processi • Acquisire il concetto di risorsa condivisa • Distinguere le richieste e le modalità di accesso alle risorse • Apprendere l'utilizzo del grafo di Holt per descrivere i processi • Conoscere la differenza tra processo e thread • Acquisire il concetto di programmazione concorrente • Acquisire il concetto di interazione tra processi • Conoscere le caratteristiche di un linguaggio concorrente 	<ul style="list-style-type: none"> • Utilizzare i thread in Java • Conoscere i metodi sleep(), yield() e join() in Java 	<ul style="list-style-type: none"> • Descrivere l'interazione processi-risorse con il grafo di Holt • Realizzare e semplificare il grafo delle precedenze • Scrivere in pseudocodifica programmi concorrenti utilizzando l'istruzione fork-join • Scrivere in pseudo codifica programmi concorrenti utilizzando l'istruzione cobegin-coend 	<ul style="list-style-type: none"> • Scrivere programmi concorrenti in Java 	<ul style="list-style-type: none"> • Informatica: linguaggio Java 	<p>Febbraio-Marzo</p>



Istituto di Istruzione Superiore "Giorgi – Woolf"

Viale Palmiro Togliatti, 1161 – 00155 Roma

<p>MODULO 5 COMUNICAZIONE E SINCRONIZZAZIONE</p>		<ul style="list-style-type: none"> • Conoscere il modello ad ambiente globale e locale • Comprendere l'esigenza di sincronizzazione • Comprendere il concetto di indivisibilità di una primitiva • Sapere il funzionamento dei semafori di Dijkstra • Avere il concetto di regione critica e mutua esclusione • Sapere la differenza tra interleaving e overleaving • Comprendere le condizioni di Bernstein • Avere il concetto di starvation e di deadlock • Comprendere le proprietà di safety, di fairness e di liveness 	<ul style="list-style-type: none"> • Risolvere le situazioni di starvation • Risolvere le situazioni di deadlock 	<ul style="list-style-type: none"> • Individuare le tipologie di errori nei processi paralleli • Definire e utilizzare i semafori di basso livello e spinlock() • Implementare i monitor di Hoare in Java 	<ul style="list-style-type: none"> • I semafori e i monitor in Java • I deadlock in Java 		<p>Aprile – Maggio</p>
<p>MODULO 6** Arduino o Raspberry Pi Interazione tra HW e SW</p>		<ul style="list-style-type: none"> • Progettare e costruire mini circuiti di base per l'interfacciamento tra la scheda Arduino ed il PC / o scheda RaspberryPi 	<ul style="list-style-type: none"> • Simulazione di circuiti elettrici programmabili. • Implementazione del codice su IDE Arduino 	<ul style="list-style-type: none"> • Arduino: • L'IDLE Arduino • Lo sketch blink. • Input digitali e comunicazioni seriali. • Input e Output analogici. • Prova di realizzazione di un progetto completo. 	<ul style="list-style-type: none"> • Uso della scheda Arduino per provare i comandi principali con integrazione dei comandi del SO / o scheda RaspberryPi 	<ul style="list-style-type: none"> • Integrazione con la materia Elettronica digitale. 	<p>Maggio – Giugno</p>

<p>OBIETTIVI MINIMI DISCIPLINARI</p>	<p>Le conoscenze relative agli obiettivi minimi sono quelle riportate nel curricolo, ma in contesti basilari ed accettabili a livello di approfondimento/difficoltà e con competenze/abilità minime o parziali.</p>	
<p>APPROCCIO DIDATTICO COMUNE A TUTTI I MODULI</p>	<p>METODOLOGIE</p>	<p>STRUMENTI</p>



Istituto di Istruzione Superiore "Giorgi – Woolf"

Viale Palmiro Togliatti, 1161 – 00155 Roma

	<ul style="list-style-type: none">• Didattica laboratoriale.• Lezione frontale.• Lezione dialogata.• Classe capovolta.• Apprendimento per scoperta.• Apprendimento per progetti.• Lavoro di gruppo• Lavoro individuale, interventi personalizzati.• Brainstorming.• Flipped classroom.	<ul style="list-style-type: none">• Compiti in classe• interrogazioni alla cattedra• interrogazioni al posto• interrogazioni programmate• questionari (prove oggettive)• relazioni da svolgere a casa• Lavagna.• Uso di PC in Laboratorio.• Piattaforma di e-learning.• Appunti elaborati dal docente.• Libro di testo.• Manuali tecnici.• Verifiche orali e scritte.• Test a risposta chiusa e aperta.• Sussidi audiovisivi.• Ambiente di sviluppo integrato (IDE).
--	---	---