

**PROGRAMMA PREVENTIVO - ANNO SCOLASTICO 2020 – 2021**

<b>DEI PROF.</b>	<b>DOCENTE DI</b>	<b>NELLA CLASSE</b>	<b>INDIRIZZO</b>	<b>ORE SETTIMANALI</b>
<b>Cannone Maria Del Vecchio Anna</b>	<b>Tecnologie e progettazione di sistemi informatici e di telecomunicazione</b>	<b>4 E</b>	<b>INFORMATICA E TELECOMUNICAZIONI</b>	<b>4 di cui 2 in codocenza</b>

<b>MODULI (TITOLO)</b>	<b>CONTENUTI</b>	<b>OBIETTIVI</b>	<b>TEMPI</b>	<b>SPAZI E MEZZI</b>	<b>COLLEGAMENTI INTER-DISCIPLINARI</b>	<b>METODI</b>	<b>CRITERI DI VALUTAZIONE</b>	<b>TIPOLOGIA DELLE PROVE</b>
<b>MODULO 1: Processi sequenziali e paralleli</b>	Il modello a processi -Stato di processi -Comandi per la creazione, sospensione e terminazione dei processi -PCB (Process Control Block) -Risorse e condivisione: -Generalità, -Classificazioni, -Grafo di Holt. -I thread o“processi leggeri” -Generalità, -Processi pesanti” e “processi leggeri”, -Soluzioni adottate: -single threading vs multithreading, -Realizzazione di thread, -Thread POSIX, -Stati di un thread, -Utilizzo dei thread. -Elaborazione sequenziale e concorrente: -Generalità, -Processi non sequenziali e grafo di precedenza, -Scomposizione di un processo non sequenziale. -La descrizione della concorrenza: -Esecuzione parallela, -Fork-join, Cobegin-coend, -Equivalenza di fork-join e cobegin-coend;	Al temine del modulo lo studente sarà in grado di utilizzare gli strumenti della programmazione per la realizzazione di programmi concorrenti utilizzando l’istruzione fork-join, l’istruzione cobegin-coend; programmi multiprocessi in linguaggio C; essere in grado di utilizzare i thread in linguaggio C, di utilizzare i thread in linguaggio Java	20	Per ogni modulo:  Laboratorio;  Appunti, libro di testo e utilizzo del computer	Per ogni modulo:  <b>Inglese:</b>  Conoscenza della terminologia tecnica in italiano ed in inglese. Saper leggere e capire documentazione tecnica di livello medio in inglese.  <b>Informatica:</b>  Linguaggi di programmazione.  <b>Sistemi e Reti:</b>  Strumenti e principi base della comunicazione in rete.	Per ogni modulo:  Lezione frontale in interazione.  Lezione di laboratorio con esercitazioni pratiche  Risorse online.  Alcune lezioni potranno essere tenute in lingua inglese.	In generale:  Il voto viene calcolato ripartendo in proporzione il punteggio riportato nella prova tra i voti 2 e 10, dunque il livello di sufficienza è il 50%, ferma restando la possibilità di successiva verifica orale su alcuni degli obiettivi non raggiunti.	Per ogni modulo:  Verifiche Scritte:  Test a scelta multipla o con domande aperte, interrogazioni. Alcune verifiche potranno essere somministrate (integralmente o parzialmente) in lingua inglese.

	-Semplificazione delle precedenze.		
Esercitazioni di laboratorio Modulo 1	L'emulatore Cygwin, L'ambiente di sviluppo Dev-C++; La fork in C, Fork annidate ed esecuzione non deterministica; Le funzioni wait() e waitpid(); Fork-join e cobegin- coend; I thread in C, Thread e parametri, Thread in ambiente Dev- cpp e linux-di sviluppo I thread in Java: concetti base, Priorità e parametri nei thread Java, I thread Java: i metodi sleep, yield e join.		20
<b>MODULO2: Comunicazione e sincronizzazione</b>	-La comunicazione tra processi -Comunicazione: -modelli software e hardware -Modello a memoria comune (ambiente globale, global environment) -Modello a scambio di messaggi (ambiente locale, message passing) -La sincronizzazione tra processi -Errori nei programmi concorrenti -Definizioni e proprietà -Proprietà non funzionali: safety e liveness; -semafori; -Semafori di basso livello e spin lock() -Semafori di Dijkstra -Semafori binari vs semafori di Dijkstra	Al temine del modulo lo studente sarà in grado di Individuare le tipologie di errori nei processi paralleli Definire e utilizzare i semafori di basso livello e spin lock() Utilizzare gli strumenti di sincronizzazione per thread in C Utilizzare le condition variable in C Implementare i monitor in C/Java Utilizzare gli strumenti di sincronizzazione per thread in C Risolvere le situazioni di starvation Risolvere le situazioni di deadlock Risolvere i problemi produttore/consumatore in C/Java	20

-Semafori e mutua esclusione  
-Mutua esclusione tra gruppi di processi  
-Semafori come vincoli di precedenza  
-Problema del rendez-vous  
-Problemi "classici" della programmazione concorrente:  
-produttori/consumatori;  
-Problema dei lettori e degli scrittori  
-Problemi "classici" della programmazione concorrente: deadlock,  
-banchiere e filosofi a cena  
-Individuazione dello stallo  
-Come affrontare lo stallo  
-Esempio classico: problema dei filosofi a cena  
-I monitor: Generalità  
-Utilizzo dei monitor  
-Variabili condizione e procedure di wait/signal  
-Emulazione di monitor con i semafori  
-Lo scambio di messaggi:  
-Generalità  
-Canali di comunicazione  
-Primitive di comunicazione  
-asimmetrica da-molti-a-uno  
-Primitive di comunicazione asimmetrica da-molti-a-molti (cenni)

Risolvere il problema dei filosofi in C/Java

<p>Esercitazioni di laboratorio Modulo 2</p>	<p>La comunicazione tra processi mediante segnali asincroni Thread e schedulazione I semafori binari in C La soluzione del deadlock dei filosofi in C con i mutex La soluzione del problema produttori/consumatori con i semafori classici Variabili condizione I monitor con le variabili condition in C I monitor con i semafori in C I semafori in Java I monitor in Java Un esempio con i Java thread: corsa di biciclette Il deadlock in Java</p>		20					
<p><b>MODULO3:</b> <b>I requisiti software</b></p>	<p>-La specifica dei requisiti -Requisiti software e stakeholder -Classificazione dei requisiti -I requisiti: l'anello debole dello sviluppo software -Raccolta e analisi dei requisiti -Tipi di raccolta dei requisiti -La fase di esplorazione -Problemi della fase di esplorazione -Attori, casi d'uso e scenari -Tipi di scenari -Descrizione dei casi d'uso -Relazioni tra casi d'uso -Documentazione dei casi d'uso -La documentazione dei requisiti</p>	<p>Al termine del modulo lo studente sarà in grado di  -Individuare i requisiti utente -Individuare i requisiti di sistema -Utilizzare le tecniche di esplorazione -Individuare gli scenari d'uso -Analizzare il documento di Specifica dei Requisiti Software (SRS) -Acquisire la struttura di un SRS -Saper descrivere in UML i casi d'uso -Saper descrivere in UML il diagramma di contesto -Saper documentare i casi d'uso -Saper compilare il documento di Specifica dei Requisiti Software</p>	20					

	-Requirements Documents proposto da Sommerville -Realizzare un efficace documento SRS	-Validare le specifiche di un SRS					
Esercitazioni di laboratorio Modulo 3	La realizzazione degli Use Case Diagram con StarUML La realizzazione degli Use Case Diagram con ArgoUML		10				

NOTE: La tempistica prevista è solo orientativa. In itinere si vedrà dove soffermarsi di più o meno e compatibilmente con le ore di lezione che saranno realmente disponibili.