INSTITUTO/S: Instituto de Tecnología e Ingeniería

CARRERA/S: Licenciatura en Informática/ Tecnicatura en Informática

MATERIA: Programación Concurrente

NOMBRE DEL RESPONSABLE DE LA ASIGNATURA: Daniel Buaon

EQUIPO DOCENTE: ---------

CUATRIMESTRE: Licenciatura: 1ro / Tecnicatura: 4to

AÑO: Licenciatura: 3ro / Tecnicatura: 2do

PROGRAMA N°: Licenciatura: 25 / Tecnicatura: 14 (**Aprob. Por Cons.Directivo fecha XX)**

Instituto/s: Instituto de Tecnología e Ingeniería

Carrera/s: Licenciatura en Informática / Tecnicatura en Informática

Nombre de la materia: Programación Concurrente

Responsable de la asignatura y equipo docente: Daniel Buaon

Cuatrimestre y año: Licenciatura: 1ro del 3er año / Tecnicatura: 4to del 2do año

Carga horaria: 4 hs semanales

Programa N°: 25

Código de la materia en SIU: 766

Programación Concurrente

1. Fundamentación

Todos los sistemas de computo moderno se componen de multiples tareas que se ejecutan en forma simultánea e independiente.

La programación concurrente comprende las técnicas para expresar la concurrencia de tareas y solución a los problemas de comunicación y sincronización entre las mismas.

Su conocimiento es fundamental para todo profesional involucrado con software.

2. Propósitos y/u objetivos

Objetivos
Que el/la estudiante:

- Tenga conocimiento de los modelos más utilizados, las técnicas y conceptos básicos asociados a la programación concurrente.
- Comprenda las dificultades asociadas a la interacción de componentes en un sistema concurrente y conozca los recursos del que dispone el programador para mitigarlos.
- Pueda analizar y desarrollar programas concurrentes.
- Diseñe e Implemente un pequeño modelo de programa concurrente con un objetivo específico.

3. Programa sintético:

Concurrencia. Los porqués de la concurrencia. Concurrencia y paralelismo. Modelo de memoria compartida, atomicidad e independencia. Manejo de memoria en ejecución. Secciones críticas, locks y barriers, semáforos, monitores y condition variables, Rendezvous. Problemas de la concurrencia: Starvation, Deadlocks, Liveness y Progress, Safety, Raceconditions, Fairness. Modelo de pasaje de mensajes: Comunicación sincrónica vs comunicación asincrónica, Modelo de transacciones. Modelos de interacción: Cliente/Servidor, Productor/Consumidor. Aplicación de los conceptos estudiados en lenguajes de programación concretos, mecanismos de sincronización.

4. Programa analítico

4.1 Organización del contenido:

Unidad 1: Conceptos básicos.

Objetivos de los sistemas concurrentes. Procesamiento secuencial, concurrente y paralelo. Características. Evolución histórica. El modelo de CSP (Communicating Sequential Processes). Programa concurrente. No determinismo. Multithreading. Concurrencia y paralelismo. Algoritmos concurrentes, distribuidos y paralelos. Relación con la arquitectura. Tendencias actuales en procesadores. Conceptos de arquitecturas Grid y Cloud. Sincronización y comunicación. Sincronización por exclusión mutua y por condición. Prioridad, granularidad, deadlock, manejo de recursos.

Unidad 2: Concurrencia y sincronización.

Aspectos de programación secuencial Especificación y semántica de la ejecución concurrente. Acciones atómicas y sincronización. El problema de interferencia. Atomicidad de grano fino y de grano grueso. La propiedad de “A lo sumo una vez”. Semántica. Técnicas para evitar interferencia. Propiedades de seguridad y vida. Políticas de scheduling y Fairness. Requerimientos para los lenguajes de programación. Problemas en sistemas distribuidos.

Unidad 3: Concurrencia con variables compartidas.

Sincronización por variables compartidas. Secciones críticas (SC). Definición del problema. Algoritmos clásicos de soluciones fair al problema de la SC (tie-breaker, ticket, bakery). Implementación de sentencias Await arbitrarias. Sincronización barrier. Definición. Planteo de soluciones (contador compartido, flags y coordinadores, árboles, barreras simétricas, butterfly). Algoritmos data parallel. Sincronización por semáforos. Sintaxis y semántica. Usos básicos y técnicas de programación. Semáforos binarios divididos (split). La técnica “passing the baton”. Sincronización por condición general. Alocación de recursos. Políticas de alocación. Sincronización por monitores. Monitores. Sintaxis y semántica. Disciplinas de señalización: “Signal and wait“ y “Signal and continue”: diferencias y efecto sobre la programación. La técnica “passing the condition”. Implementaciones Conceptos de implementación de procesos en arquitecturas mono y multiprocesador. Kernel monoprocesador y multiprocesador.

Unidad 4: Programación distribuida.

Concurrencia con pasaje de mensajes Programas distribuidos. Definición. Relación entre mecanismos de comunicación. Clases básicas de procesos: productores y consumidores, clientes y servidores, peers. Control de concurrencia en Sistemas Distribuidos. Mensajes asincrónicos Sintaxis y semántical. Peers. Intercambio de valores. Mensajes sincrónicos Sintaxis y semántica. Ejemplos Remote Procedure Calls y Rendezvous. Sintaxis y semántica. Similitudes y diferencias. RPC: Sincronización en módulos. RPC en lenguajes reales. RMI. Ejemplos.

Unidad 5: Paradigmas de interacción entre procesos distribuidos.

Resolución de problemas mediante diferentes paradigmas de interacción entre procesos: Servidores replicados, heartbeat, pipeline, prueba-eco, broadcast, token passing, manager/workers. Objetivos del cómputo paralelo. Necesidad del paralelismo. Areas de aplicación. Computación científica. Diseño de algoritmos paralelos.

4.1 Bibliografía y recursos obligatorios:

Raynal, M. (2012). *Concurrent Programming: Algorithms, Principles, and Foundations*. USA: Springer.

Lea, D. (1999). Concurrent Programming in Java: Design Principles and Patterns. USA: Addison-Wesley Professional

Naiouf, M.; De Giusti, A.; De Giusti, L. y Chichizola, F. (2012). *Conceptos de Concurrencia y Paralelismo*. La Plata, Argentina: EDULP.

4.2 Bibliografía optativa:

Taubenfeld, G. (2006). *Synchronization Algorithms and Concurrent Programming*. USA: Prentice Hall.

Downey, A. (2007). *The Little Book of Semaphores, Second Edition. Free book*. Disponible en http://www.freetechbooks.com/the-little-book-of-semaphores-second-edition-t519.html

5. Metodologías de enseñanza:

Las clases se organizan como teórico-prácticas. Estas clases podrán ser tanto presenciales como virtuales en forma indistinta.

Durante las clases se presenta el contenido formal y se ahonda en las características de la programación concurrente. Las clases son acompañadas por un apunte en forma de diapositivas que sirve a los alumnos/as como material de referencia más allá de la bibliografía sugerida.

Después de completado cada tema se resuelven ejercicios de programación haciendo hincapié en las características de la programación concurrente mediante las computadoras del laboratorio o las computadoras personales de los alumnos.

En la práctica se plantean ejercicios que pueden ser realizados como trabajo experimental sobre arquitecturas multiprocesador distribuidas (clusters), multiprocesadores con memoria compartida e híbridos

El objetivo de las prácticas es completar la cursada con la implementación de un pequeño proyecto cuyo diseño se discute sobre la base de la teoría.

Plan de trabajo en el campus:

El Campus Virtual es un espacio fundamental para el desarrollo de la asignatura. En el aula virtual se propondrá material educativo, apuntes de clase, bibliografía, así como también el programa y cronograma de la asignatura y las guías de Trabajos Prácticos y ejercicios.
Se establecerá también un canal de Slack para comunicación para consultas y anuncios.

6. Actividades de investigación y extensión (si hubiera)

7. Evaluación y régimen de aprobación

7.1 Aprobación de la cursada

Consistirá en dos exámenes parciales con recuperatorios, según el cronograma previsto, de la totalidad de la materia descripta en el programa. Se contemplará también como nota de evaluación el trabajo práctico final de implementación.

Para aprobar la cursada y obtener la condición de regular, el régimen académico establece que debe obtenerse una nota no inferior a cuatro (4) puntos y poseer una asistencia no inferior al 75% en las clases presenciales.

Podrán acceder al recuperatorio solo aquellos/as estudiantes que hayan obtenido una nota inferior o igual a 6 (seis) puntos en el examen parcial. La calificación que los/as estudiantes obtengan reemplazará la calificación obtenida en el examen que se ha recuperado y será la considerada definitiva a los efectos de la aprobación.

7.2 Aprobación de la materia

La materia puede aprobarse por promoción, evaluación integradora, examen final o libre.

Promoción directa: tal como lo establece el art°17 del Régimen Académico, para acceder a esta modalidad, el/la estudiante deberá aprobar la cursada de la materia con una nota no inferior a siete (7) puntos, no obteniendo en ninguna de las instancias de evaluación menos de seis (6) puntos, sean evaluaciones parciales ; recuperatorios o Trabajo práctico. El promedio estricto resultante deberá ser una nota igual o superior a siete (7) sin mediar ningún redondeo.

Evaluación integradora: tal como lo establece el art°18 del Régimen Académico, podrán acceder a esta evaluación aquellos/as estudiantes que hayan aprobado la cursada con una nota de entre cuatro (4) y seis (6) puntos.
La evaluación integradora tendrá lugar por única vez en el primer llamado a exámenes finales posterior al término de la cursada. Se realizará el mismo día y horario de la cursada y será administrado, preferentemente, por el/la docente a cargo de la comisión. Se aprobará tal instancia con una nota igual o superior a cuatro (4) puntos, significando la aprobación de la materia. La nota obtenida se promediará con la nota de la cursada.

Examen final: Instancia destinada a quienes opten por no rendir la evaluación integradora o hayan regularizado la materia en cuatrimestres anteriores. Son presenciales, individuales, sin material a la vista. La devolución de los resultados se realiza el mismo día en forma personalizada. Se evalúa la totalidad de los contenidos del programa de la materia y se aprueba con una calificación igual o superior a cuatro (4) puntos. Esta nota no se promedia con la cursada.

7.3 Criterios de calificación

La calificación de cada evaluación se determinará en la escala 0 a10, con los siguientes valores: 0, 1, 2 y 3: insuficientes; 4 y 5 regular; 6 y 7 bueno; 8 y 9 distinguido; 10 sobresaliente.

8. Cronograma

Clase	Semana	Teoría
1	1/4	Introducción, conceptos basicos. Concurrencia con Python
2	8/4	Concurrencia con variables compartidas Exclusión Mutua - Secciones críticas (SC). Locks - Deadlocks
3	22/4	Problemas Clasicos de Programacion Concurrente Mutex, Locks, RLocks y Condition
4	29/4	Problemas Clasicos de Programacion Concurrente Semáforos
5	6/5	REPASO Y ACTIVIDADES PRATICAS
6	13/5	PRIMERA EVALUACION
7	20/5	Monitores
8	27/5	Mensajes
9	3/6	Colas de procesos
10	10/6	Programacion Asincronica
11	24/6	Programacion por eventos
12	1/7	REPASO Y ACTIVIDADES PRATICAS

13	8/7	SEGUNDA EVALUACION
14	15/7	RECUPERATORIOS