

Información del curso

Pedro O. Pérez M., PhD

Multiprocesadores
Tecnológico de Monterrey

pperezm@tec.mx

07-2020

Contenido I

Información del profesor

Información del profesor

Información del curso

Intenciones educativas

Objetivos generales

Metodología

Evaluación

Normas de clase

Bibliografía

Contenido II

Herramientas a utilizar

Herramientas a utilizar

Programación en parejas (Pair programming)

¿Qué es? ¿Cómo funciona?

Información del profesor

- ▶ Pedro Oscar Pérez Murueta
 - ▶ ISC Mayo 1994
 - ▶ MTI Mayo 2002
 - ▶ DCC Diciembre 2019
- ▶ Correo: pperezm@tec.mx
- ▶ Oficina: Edificio 2, Piso 3
- ▶ Horario de asesoría: Se encuentra en la puerta de mi oficina.



Intenciones educativas

- ▶ Curso teórico de nivel avanzado en programación de equipo de cómputo que proporciona a los estudiantes los conocimientos sobre el funcionamiento de sistemas de cómputo basados en microprocesadores de núcleos múltiples y de arquitecturas de múltiples microprocesadores interconectados.
- ▶ Requiere de conocimientos previos de sistemas operativos, interfaces de equipo de cómputo. Como resultado del aprendizaje el alumno podrá diseñar y codificar algoritmos utilizando el paradigma de fragmentación de tareas para resolver problemas usando sistemas de cómputo de núcleos múltiples y/o sistemas con múltiples procesadores.

Objetivos generales

Al finalizar el curso el alumno será capaz de comprender el funcionamiento de un microprocesador, su arquitectura interna y sus técnicas de programación para la codificación de algoritmos paralelos, analizando la eficiencia de sus implementaciones, mediante herramientas de evaluación de desempeño.

Metodología

- ▶ **Autoestudio:** Cada semana se deberá realizar un autoestudio previo. Los autoestudios consistirán generalmente en la lectura de un capítulo de alguno de los libros de texto.
- ▶ **Actividad colaborativa:** Estas actividades reforzarás lo visto en el autoestudio. En equipos colaborativos, y usando la técnica de Pair Programming, deberás implementar una solución paralela eficiente al problema presentado.
- ▶ **Exámenes semanales:** Cada semana, al inicio de la primera sesión se aplicará un examen semanal. El examen dura 30 minutos, es de opción múltiple y cubrirá los temas vistos en la semana previa. Tendrás dos oportunidades para contestar el examen. Tendrás dos oportunidades para resolver esta actividad.

- ▶ **Exámenes semanales:** Cada semana, al inicio de la primera sesión se aplicará un examen semanal. El examen dura 30 minutos, es de opción múltiple y cubrirá los temas vistos en la semana previa. Tendrás dos oportunidades para contestar el examen.
- ▶ **Foros:** ¿En qué consiste esta actividad? Debes realizar un comentario sobre artículos de interés relacionados con Multiprocesadores, además de comentar la aportación de un compañero del grupo. Importante: para que la actividad se considere completa, debes realizar las dos aportaciones.

- **Artículo de investigación:** La actividad final del curso consiste en escribir un artículo de investigación en donde se resuelva un problema en el que se utilicen y comparen diferentes tecnologías de programación paralela y concurrente.

Evaluación

Evaluación parcial		Evaluación final	
Exámenes semanales	100 %	Exámenes semanales	30 %
		Actividades colaborativas	35 %
		Artículo de investigación	25 %
		Foros	10 %

Normas de clase

Exámenes

- ▶ Los exámenes podrán ser presentados solamente en la fecha estipulada. El no presentar un examen implica una calificación de NP (No Presentó).

Tareas y Proyectos

- ▶ Toda tarea y/o proyecto tendrá su fecha y horario de entrega que es inamovible. Vencido el término de entrega no se recibirán tareas y/o proyectos.
- ▶ Todas las tareas son individuales a menos que explícitamente se pida trabajar en grupo.

Redacción y Organización

- ▶ La mala redacción, organización y ortografía en la elaboración de tareas, proyectos, presentaciones y exámenes, será causa de penalización en la calificación correspondiente.

Asistencia a clases

En lo que respecta a esta clase:

- ▶ La sesión de clase inicia 5 minutos después del horario establecido (16:05). Si no estás al inicio de la misma, se considerará que no asististe a esa sesión. Asimismo, también se considera inasistencia si te retiras, sin permiso del profesor, antes de terminar la sesión de clase.
- ▶ No podrás acreditar, bajo ningún concepto, las actividades (tareas y/o exámenes) de las sesiones a las cuales no hayas asistido. Además, será tu responsabilidad estudiar el material visto en esas sesiones.

Calificaciones

- ▶ Las calificaciones parciales y final se expresan en escala de uno a cien.
- ▶ La calificación mínima aprobatoria es 70 (SETENTA).






Faltas a la Integridad Académica en Tareas, Proyectos o Exámenes

- ▶ Las faltas a la integridad académica, como la copia o tentativa de copia en cualquier tipo de examen o actividad de aprendizaje; el plagio parcial o total; facilitar alguna actividad o material para que sea copiada y/o presentada como propia; la suplantación de identidad; falsear información; alterar documentos académicos; vender o comprar exámenes o distribuirlos mediante cualquier modalidad; hurtar información o intentar sobornar a un profesor o cualquier colaborador de la institución; entre otras acciones más son consideradas faltas grave. Cuando un alumno cometa un acto contra la integridad académica, se le asignará una calificación reprobatoria a la actividad, examen, período parcial o final. La calificación reprobatoria asignada por el profesor será inapelable, y a esta sanción se sumarán las otras posibles que determine el Comité de Integridad Académica de Campus. Esto tal como lo indica el Reglamento Académico en su CAPÍTULO IX: Faltas a la integridad académica.

Bibliografía

Bibliografía

Libros de Texto

	[AKHTER] Shameem Akhter, Jason Roberts. Multi-Core Programming: Increasing Performance through Software Multi-threading. Intel Press, 2006.
	[BRESHEARS] Clay Breshears. The Art of Concurrency. O'Reilly, 2009.
	[CESARINI] Francesco Cesarini, Simon Thompson. Erlang Programming. O'Reilly, 2009.
	[GOETZ] Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, Doug Lea. Java Concurrency in Practice. Addison-Wesley, 2006.
	[PACHECO] Peter Pacheco. An Introduction to Parallel Programming. Morgan Kaufmann, 2011.



SCAN ME

<https://itesm.zoom.us/my/pperezm>



SCAN ME

<https://shorturl.at/ertE1>



SCAN ME

<https://shorturl.at/ekoQ9>



[https://github.com/Manchas2k4/
multiprocessors](https://github.com/Manchas2k4/multiprocessors)



[https:
//www.remind.com/join/ecf28b](https://www.remind.com/join/ecf28b)

¿Qué es? ¿Cómo funciona?

- ▶ Dos programadores trabajan juntos, uno al lado del otro frente a una sola computadora.
- ▶ Ambos colaboran juntos en un mismo diseño, algoritmo, código o prueba.
- ▶ En todo momento existen dos roles:
 - ▶ **El conductor** tiene el control del lápiz/teclado/mouse, y activamente implementa el programa.
 - ▶ **El navegante** continuamente y activamente examina el trabajo del conductor detectando defectos tácticos (sintaxis, convenciones de codificación, etc.), pensando en alternativas, buscando recursos, considerando implicaciones estratégicas del trabajo en cuestión y haciendo preguntas.

- ▶ Cuando la pareja lo determine apropiado, pueden realizar una "lluvia de ideas" para resolver de manera conjunta las dificultades que se susciten.
- ▶ El lápiz/teclado/mouse debe deslizarse de un lado a otro de manera periódica para que los roles puedan intercambiarse.
- ▶ Los dos programadores son responsables por igual del éxito o fracaso del producto.
- ▶ Requiere de más esfuerzo y concentración debido a que el ritmo es forzado por la otra persona todo el tiempo. Ninguna de las dos personas puede reducir su paso.

¿Qué necesitamos instalar?

- ▶ Windows: Ubuntu - Windows Subsystem (dir)
- ▶ Atom (<https://atom.io/>) o Sublime Text (<https://www.sublimetext.com/>)