Información del curso

Pedro O. Pérez M., PhD.

Análisis y diseño de algoritmos
Tecnológico de Monterrey

pperezm@tec.mx

01-2020

Contenido

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.

- Github: `https://github.com/Manchas2k4/algorithms`
- Remind: `https://www.remind.com/join/k7hgk2`

Intenciones educativas

Este es un curso de nivel intermedio de computación en donde conocerás técnicas de diseño y análisis de algoritmos. Requiere de conocimientos previos de matemáticas discretas, programación en un lenguaje de alto nivel, estructuras de datos. Como resultado de tu aprendizaje se espera que seas capaz de analizar y comparar algoritmos, emplear técnicas de diseño de algoritmos, diseñar algoritmos que resuelvan problemas específicos de manera eficiente, conocer de la clasificación de algoritmos y del problema P vs NP.

Objetivos generales

Al finalizar el curso, podrás analizar algoritmos para demostrar su efectividad y la complejidad temporal. Comprenderás, compararás y aplicará las diferentes estrategias y técnicas de diseño de algoritmos.

Metodología

Para alcanzar los objetivos que persigue la materia hemos desarrollado una estrategia organizada de la siguiente manera:

- **Sesiones de clase:** Se verán distintos tópicos relacionados con el análisis y diseño de algoritmos.
- **Actividad colaborativa:** Cada dos semanas, el profesor te presentará dos problemas. En equipos colaborativos, y usando la técnica de Pair Programming, deberás selecciona uno de los problemas, diseñar e implementar una solución eficiente para el mismo.

- **Concurso de programación:** Cada dos semanas, se realizará un concurso de programación. Cada concurso constará de tres problemas: un problema fácil, uno medio y uno difícil. Cada problema tendrás un valor en puntos: 1 (fácil), 2 (medio) y 3 (difícil). Deberás resolver, **de manera individual**, resolver al menos un problema. La calificación asignada a esta actividad se dará en relación con los puntos acumulados. Para obtener la calificación máxima, deberás acumular 27 puntos a lo largo de los concursos.

- **Foros:** En los foros se discutirán algunos artículos de interés relacionados con algoritmos.
- **Exámenes de tema:** Se te presentarán algunos problemas no estructurados de los cuales deberás desarrollar un algoritmo eficiente.
- **Examen final:** El examen final cubrirá todos los temas vistos durante el semestre. Se te presentarán pseudocódigos para que calcules la función polinomial de complejidad o problemas no estructurada para los cuales deberás desarrollar un algoritmo eficiente.

Evaluación

Evaluación parcial	
Actividades colaborativas	30 %
Exámenes de tema	70 %

Evaluación final	
Actividades colaborativas	20 %
Concursos de programación	15 %
Foros	10 %
Exámenes de tema	35 %
Examen final	20 %

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).
- El cambio de fecha de algún examen parcial deberá realizarse, a petición de los estudiantes, durante las dos primeras semanas de clase. Éste se hará sólo si se cuenta con el consenso del grupo y del profesor.

Asistencia a clases

En lo que respecta a esta clase:

- La sesión de clase inicia 5 minutos después del horario establecido (10:05). 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.

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.

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	
	**[JARAMILLO]** Jaramillo, Eduardo y Guerrero, Luz. Análisis y diseño de algoritmos: un enfoque práctico. Universidad de Colombia, 2016.
	**[BENOIT]** Benoit, Anne; Robert, Yves; Vivien, Frederic. A guide to algorithm design: paradigms, methods and complexity analysis. CRC, 2014.
	**[DEITEL]** Deitel, Harvey M. y Paul J. Deitel. C++ How to program. Cuarta Edición. Prentice Hall, 2003.

¿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.