COM-11305 Programación Competitiva
Horas por semana: 3
Créditos: 6

Objetivo:
Para lograr un buen resultado en las competencias de programación y para mejorar la calidad en el desarrollo de software es necesario conocer las técnicas y las herramientas utilizadas para las situaciones que comúnmente se plantean durante los concursos y las diferentes soluciones que han sido altamente estudiadas.

Esta materia integra los conocimientos formales y herramentales adquiridos en los cursos previos de Algoritmos y Programas, Estructuras de Datos, Álgebra y temas de desarrollo de aplicaciones por medio de programación orientada a objetos yen general de desarrollo de aplicaciones, teoría de los número y análisis de algoritmos. El curso es teórico-práctico con el propósito de conocer y dominar las técnicas en uso y los soportes tecnológicos actuales para dar soluciones completas. El temario está basado en el libro "**Competitive programming 2** : this increases the lower bound of programming contest : again" de los autores Steven Halim y Felix Halim.

Temario:

Capítulo	Tema	Conceptos
1	**Introducción**	▪ Inmersión a los concursos de programación ▪ **Tips** para ser un programador competitivo o Identificar rápidamente los tipos de problemas o Hacer un rápido análisis de complejidad o Conocer bien los lenguajes: C/C++ y Java o Crear casos de prueba para probar una solución antes de ser enviada o Evitar penalizaciones o Práctica, práctica y más práctica o Trabajo en equipo ▪ Problemas Ad-hoc
2	**Estructuras de Datos y Librerías**	▪ Librerías con estructuras de datos implementadas ▪ Estructuras de datos lineales ▪ Estructuras de datos no lineales ▪ Nuestras propias estructuras de datos o Grafos o Union-Find y conjuntos disjuntos (Disjoint Sets) o Segment Tree o Binary Indexed Tree (**BIT)**
3	**Técnicas para resolver problemas**	1. **Búsquedas** 2. **Divide y vencerás - Divide and Conquer** (D&C) a. Bisección b. Búsqueda binaria 3. **Algoritmos glotones (Greedy)** 4. **Programación dinámica - Dynamic Programming** (DP)

		a. O(n log k) for LIS b. Knapsack c. Subset Sum d. TSP
4	**Teoría de Grafos**	▪ **Recorrido de Grafos** o Depth First Search (DFS) o Breadth First Search (BFS) o Componentes conexas (Undirected Graphs) o Flood Fill - Labeling/Coloring Connected Components o Ordenamiento topológico (Directed Acyclic Graph) o Bipartite Graph Check o Graph Edges Property Check via DFS Spanning Tree o Finding Articulation Points and Bridges (in an Undirected Graph) o Finding Strongly Connected Components (in a Directed Graph) ▪ **Árbol de expasión mínimo (Minimum Spanning Tree)** o Kruskal's Algorithm o Prim's Algorithm ▪ **Caminos más cortos con un solo origen (Single-Source Shortest Paths)** o SSSP on Unweighted Graph o SSSP on Weighted Graph o SSSP on Graph with Negative Weight Cycle ▪ **Caminos más cortos entre todos los nodos (All-Pairs Shortest Paths)** o Floyd Warshall's DP Solution ▪ **Maximum Flow** o Ford Fulkerson's Method o Edmonds Karp's ▪ **Grafos especiales** o Directed Acyclic Graph o Trees o Eulerian Graph o Bipartite Graph

5	**Matemáticas**	▪ **Clase Big Integer de Java** ▪ **Combinatoria** o Fibonacci Numbers o Binomial Coefficients o Catalan Numbers ▪ **Teoría de Números** o Prime Numbers o Greatest Common Divisor (GCD) & Least Common Multiple (LCM) o Factorización de eficiente de números o Uso de factores primos o Funciones que utilizan factores primos o Aritmética modular o Extended Euclid: Solving Linear Diophantine Equation ▪ **Probability Theory** ▪ **Cycle-Finding** o Solution using Efficient Data Structure o Floyd's Cycle-Finding Algorithm ▪ **Teoría de Juegos** o Árboles de decisión o Conceptos matemáticos para encontrar una solución constante O(1) o Juego del Nim ▪ **Exponenciación de matrices** o Exponenciación biaria (logarítmica) o SquareMatrix Exponentiation
6	**Procesamiento de Cadenas**	▪ **StringMatching** o Knuth-Morris-Pratt (KMP) Algorithm o StringMatching in a 2D Grid ▪ String Processing con programación dinámica o String Alignment (Edit Distance) o Longest Common Subsequence o Palindrome ▪ **Suffix Trie/Tree/Array** o Suffix Trie o Suffix Tree o Suffix Array

7	**Geometría Computacional**	▪ Objetos geométricos básicos o 0D: Puntos o 1D: Líneas o 2D: Círculos o 2D: Triángulos o 2D: Cuadriláteros o 3D: Esferas o 3D: Otros ▪ **Polígonos** o Representación o Perímetro de un polígono o Área de un polígono o Checking if a Polygon is Convex o Checking if a Point is Inside a Polygon o Cutting Polygon with a Straight Line o Finding the Convex Hull of a Set of Points
8	**Técnicas Avanzadas**	▪ **Descomposición de problemas** o Dos problemas en uno: ▪ Búsqueda binaria + (Greedy ó función ó simulación) ▪ SSSP y DP ▪ Varios algoritmos sobre grafos ▪ Varias técnicas matemáticas o Tres problemas en uno: ▪ Factores primos, DP, Búsqueda binaria ▪ Simulación completa, búsqueda binaria, Greedy ▪ **Técnicas avanzadas** o Informed Search: A* o Depth Limited Search o Iterative Deepening Search o Iterative Deepening A* (IDA*) ▪ **Técnicas avanzadas de programación dinámica** o DP + bitmask o Chinese Postman/Route Inspection Problem o Compilation of Common DP States o Correcta representación de estados o Minimización de parámetros en los estados o Offset Technique
9	**Algoritmos Exóticos**	▪ **Isomorfismo de grafos (Tree Isomorphing)**

Metodología y enfoque:
El profesor expone los temas y explica las ideas principales. A partir de este conocimiento se debe desarrollar código que implemente el tema y después resolver una gran cantidad de problemas que utilicen el tema aprendido.
Se trabajará indistintamente con los lenguajes C/C++ y Java.

Material en línea y agenda del curso

Semana	Material	Capítulo
01	week01_introduction.pdf ; skillset.xls ; and paper on CS32333	Ch 1
02	week02_ds_libraries.pdf	Ch 2
03	week03_paradigms.pdf	Ch 3.1-3.4
04	week04_dp_1.pdf	Ch 3.5
05	week05_graph_1.pdf	Ch 4 (except Ch 4.6 + 4.7.4)
06	week06_dp_2.pdf	Ch 3.5, 4.7.1, 5.4, 5.6, 6.5, 8.4
/	RECESS WEEK	Permite asimilar conceptos
07	Concurso intersemestral	
08	week08_graph_2.pdf	Ch 4.6 + 4.7.4
09	week09_maths.pdf	Ch 5
10	week10_string.pdf	Ch 6
11	week11_geometry.pdf	Ch 7
12	week12_harder_stuffs.pdf	Ch 8
13	Concurso Final	(Ch 1 - 8)

Evaluación: Se realizarán varios concursos para reforzar los temas y también se asignará una lista de ejercicios disponibles en la bibliografía principal. Los problemas serán revisados por medio de un juez automático en línea. Dependiendo del porcentaje de problemas resueltos de la lista de problemas propuestos, se asignará una calificación.

Bibliografía y recursos didácticos:

- **Competitive programming 2** : this increases the lower bound of programming contest : again / Steven Halim ; Felix Halim., [Singapore : National University of Singapore], 2011.
- ***uva.onlinejudge****.org* **Es necesario que todos creen una cuenta en dicha página. Cada cuenta será utilizada para monitorear el avance de cada persona.**
- **Introduction to algorithms** / Thomas H. Cormen ... [et al.]. , 3rd ed. , Cambridge, Mass. : MIT Press, c2009.
- **Programming pearls** / Jon Louis Bentley., 2nd ed. New York : ACM Press : Addison-Wesley, c2000.
- **Computational geometry : algorithms and applications** / Mark de Berg ... [et al.]., Algorithms and applications., , 2nd rev. ed., Berlin : Springer, c2000.
- **The art of computer programming**, Knuth, Donald Ervin, 1938, 2nd. ed.., Reading, Mass. : Addison-Wesley, 1973.
- **Graphs, networks and algorithms** / Dieter Jungnickel ; tr. from the German by Tilla Schade., LinkJungnickel, Dieter, Berlin : Springer, 1999.