CIENCIA DE LA COMPUTACIÓN

PROGRAMACIÓN COMPETITIVA

4 CRÉDITOS

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

CE2A

ÍNDICE

UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA
SILABO 2021-1

1. ASIGNATURA

CS3101 - Programación Competitiva

2. DATOS GENERALES

2.1 Ciclo: 6°
2.2 Créditos: cuatro (4) créditos
2.3 Horas de teoría**: dos (2) semanales**
2.4 Horas de práctica: cuatro (4) semanales
2.5 Duración del período: dieciséis (16) semanas
2.6 Condición:

- Obligatorio para Ciencia de la Computación (Eliminar si no aplica y ajustar al índice)

2.7 Modalidad: Virtual
2.8 Requisitos:

-CS2102-Análisis y Diseño de Algoritmos

3. PROFESORES

3.1 Profesor coordinador del curso

Juan Gutiérrez (jgutierreza@utec.edu.pe)
Horario de atención: Martes 6pm-7pm

3.2. Profesor(es) instructor(es) del curso

Juan Gutiérrez (jgutierreza@utec.edu.pe)
Horario de atención: Martes 6pm-7pm

4. INTRODUCCIÓN AL CURSO

La Programación Competitiva combina retos para solucionar problemas con el añadido de poder competir con otras personas. Enseña a los participantes a pensar más rápido y desarrollar habilidades para resolver problemas. Este curso enseñará la resolución de problemas algorítmicos de manera rápida combinando la teoría de algoritmos y estructuras de datos con la práctica la solución de los problemas.

Este curso enseña las técnicas y habilidades necesarias para resolver problemas de concursos de programación competitiva como los que aparecen en el ACM

ICPC, Codeforces y Topcoder. También aprenderá a pensar en algoritmos y estructuras de datos de una manera más audaz, porque muchos de los problemas requieren idear un nuevo algoritmo, basado en los algoritmos clásicos que se conocen. Estas habilidades serán de gran valor en sus entrevistas de trabajo y carrera profesional.

5. OBJETIVOS

Sesión 1. Repasar sintaxis de C++, dominar técnicas recursivas de backtracking

Sesión 2. Conocer los conceptos básicos de teoría de números. Aplicar la sieva de eratóstenes en problemas específicos. Aplicar conceptos básicos de combinatoria en problemas específicos

Sesión 3. Reconocer cuándo un problema requiere de algoritmo voraz y saber implementarlo. Aplicar técnicas de sliding window y sweep line. Dominar búsqueda binaria y sus implementaciones. Saber aplicar búsqueda ternaria y sus ventajas respecto a búsqueda binaria.

Sesión 4. Dominar programación dinámica a nivel básico. Conocer y aplicar técnicas avanzadas de programación dinámica

Sección 5. Saber construir un suffix array para responder consultas en cadenas. Dominar técnicas de strings para problemas específicos

Sección 6. Saber distinguir entre queries estáticos y dinámicos. Conocer cómo se construye un fenwick tree y un segment tree. Saber aplicar dichas estructuras a problemas específicos.

Sección 7. Dominar BFS y DFS, así como su aplicación a problemas específicos. Aplicar técnicas de programación dinámica en grafos dirigidos acíclicos. Aplicar algoritmos en componentes fuertemente conexos en casos específicos.

6. COMPETENCIAS Y CRITERIOS DE DESEMPEÑO

Los criterios de desempeño que se van a trabajar en este curso son:

1.3. Aplica conocimientos de computación apropiados para la solución de problemas definidos y sus requerimientos en la disciplina del programa (Nivel 3).

2.2. Formula problemas complejos de computación y otras disciplinas relevantes en el dominio (Nivel 3).

3.1. Diseña y evalúa sistemas, componentes o procesos que satisfacen las necesidades específicas (Nivel 3).

9.2. Demuestra habilidades relacionadas al desarrollo profesional continuo (mejora continua) (Nivel 2).

7. RESULTADOS DE APRENDIZAJE

Al final del curso de Programación Competitiva se espera que el estudiante sea capaz de:

RA1. Argumentar la complejidad algorítmica de un algoritmo eficiente usando las propiedades y técnicas conocidas de análisis de algoritmos

RA2. Construir algoritmos eficientes para problemas de programación competitiva usando las técnicas algorítmicas más adecuadas al problema

RA3. Construir programas que satisfacen las restricciones de tiempo y memoria de un problema dado

RA4. Desarrollar habilidades de adaptación a nuevas técnicas y nuevos desafíos de programación que ocurren en la construcción de software

8. TEMAS

1. **Introducción**
   - 1.1. Entrada y salida, tipos de datos, aritmética modular
   - 1.2. Algoritmos recursivos: subconjuntos, permutaciones y backtracking
2. **Matemática**
   - 2.1. Teoría de números: números primos, sieva de eratóstenes, algoritmo de euclides, exponenciación modular
   - 2.2. Combinatoria: coeficientes binomiales, números de Catalán, lema de Burnside
3. **Búsquedas y ordenamientos**
   - 3.1. Repaso de algoritmos voraces
   - 3.2. Sweep line, Sliding window
   - 3.3. Búsqueda binaria, Búsqueda ternaria
4. **Programación dinámica**
   - 4.1. Problemas clásicos: Monedas, LIS, LCS, Edit Distance, Mochila
   - 4.2. Técnicas avanzadas: bitmask, optimizaciones, uso de recurrencias y matrices
5. **Cadenas**
   - 5.1. Árboles de sufijos
   - 5.2. Algoritmo KMP
   - 5.3. Función Z
   - 5.4. Hashing
6. **Consultas de rango (Range Queries)**
   - 6.1. Consultas en arreglos estáticos
   - 6.2. Sparse Table

6.3. Binary Indexed Tree (Fenwick tree),
6.4. Segment tree
6.5. SQRT decomposition

7. Algoritmos en grafos

7.1. Repaso de temas clásicos: búsquedas, caminos mínimos, spanning trees
7.2. Árboles: caminos máximos, ancestros, queries
7.3. Aplicaciones de DFS: puentes, articulaciones, biconectividad, subgrafos eulerianos
7.4. Grafos dirigidos acíclicos: ordenación topológica, programación dinámica
7.5. Grafos dirigidos fuertemente conexos: Algoritmo de Kosaraju, 2SAT
7.6. Flujo máximo en grafos. Emparejamiento máximo en grafos bipartidos.

9. PLAN DE TRABAJO

9.1 Metodología

Se aplicarán metodologías activas como aula clase invertida y aprendizaje práctico para fomentar la participación individual y en equipo, y además para exponer las ideas por parte de los estudiantes, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso. El uso de herramientas online permitirá al alumno acceder a la información del curso, interactuar fuera del aula con el profesor y con los otros estudiantes.

9.2 Sesiones de teoría

Las sesiones teóricas serán desarrolladas bajo la estructura de clase invertida, lo que significa que el estudiante es responsable por su aprendizaje y preparación para la sesión de clase. Antes de cada clase, los estudiantes tendrán asignado un problema (indicada por el docente) y sobre dicho ejercicio se desarrollará el tema de la clase.

9.3 Sesiones de práctica (laboratorio o taller)

Las sesiones prácticas/laboratorio se desarrollarán a través de una metodología activa generando el aprendizaje práctico por parte del estudiante.

Las sesiones de práctica se caracterizan por el desarrollo de problemas de programación competitiva usando jueces virtuales, participando en concursos virtuales los alumnos podrán medir sus conocimientos adquiridos en la teoría.

10. SISTEMA DE EVALUACIÓN

EVALUACIÓN	TEORÍA	PRÁCTICA Y/O LABORATORIO
*Para aprobar el curso es necesario aprobar tanto la parte teórica como la parte práctica	Examen **E1** (20%) Examen **E2** (20%)	Evaluación continua **C1** (10%) Evaluación continua **C2** (10%) Evaluación continua **C3** (10%) Evaluación continua **C4** (10%) Evaluación continua **C5** (10%) Evaluación continua **C6** (10%)
	40%	60%
	100%	

Las rúbricas que permitirán medir las actividades más significativas del curso y que, además se relacionan con la evaluación de las competencias del estudiante son: enlace

11. SESIONES DE APOYO O TUTORÍAS

Este apartado permite formalizar los espacios de apoyo a los estudiantes y que éstos tengan la atención NECESARIA y el tiempo disponible para presentar sus dudas y consultas acerca del curso:

Semana	Fecha/ Hora	Tema a tratar	Objetivos de la sesión
2	27/04/2021 6:00 PM – 7:00 PM	Evaluaciones	Resolver dudas sobre sistema de evaluación y temas a tratar en el curso
4	11/05/2021 6:00 PM – 7:00 PM	Tópicos del curso	Resolver dudas sobre primeras 3 semanas
6	25/05/2021 6:00 PM – 7:00 PM	Tópicos del curso	Resolver dudas sobre primeras 5 semanas

8	08/06/2021 6:00 PM – 7:00 PM	Parcial	Reclamos examen parcial
10	22/06/2021 6:00 PM – 7:00 PM	Tópicos del curso	Resolver dudas sobre temas tratados en clase
12	06/07/2021 6:00 PM – 7:00 PM	Tópicos del curso	Resolver dudas sobre últimas semanas
14	20/07/2021 6:00 PM – 7:00 PM	Final	Consultas sobre ultimas notas y examen final

12. REFERENCIAS BIBLIOGRÁFICAS

- Antti Laaksonen. Guide to Competitive Programming: Learning and Improving Algorithms Through Contests. Springer 2018
- Steven Halilm, Felix Halim. Competitive Programming 3: The New Lower Bound of Programming Contests, Volumen3. Lulu.com, 2013
- Steven S. Skiena, Miguel A. Revilla. Programming Challenges: The Programming Contest Training Manual. Springer Science & Business Media, 2006
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2009). *Introduction to algorithms*. MIT press.