ÁREA 1. ALGORITMIA

SUBÁREA 1.1 ANÁLISIS Y DISEÑO DE ALGORITMOS

BIBLIOGRAFÍA DE LA GUÍA:

- Aho, Alfred, Hopcroft, John, E. y Ullman, Jefrey. (1988). Estructuras de Datos y Algoritmos. Estados Unidos: Addison-Wesley Iberoamericana. 413 p.
- Brassard, G. y Bratley, P. (2008). Fundamentos de Algoritmia. España: Pearson, Prentice Hall.
- Cairo, Osvaldo. (2005) Metodología de la Programación. Algoritmos, diagramas de flujo y programas. 3ª ed. México: Alfaomega.
- Cormen, Thomas, H. et al. (2009). Introduction to Algorithms. 3a. ed. Estados Unidos: The MIT Press. 1 313 p.
- Corona Nakamura, María Adriana y Ancona Valdez, María de los Ángeles (2011). Diseño de algoritmos y su codificación en lenguaje C. México: McGraw-Hill.
- Kernighan W., Brian. y Dennis, Ritchie. (1995). Lenguaje de Programación C. 2a ed. México: Pearson. 153 p.
- Wirth, Niklaus. (1987). Algoritmos y estructuras de datos. 2a ed. México: Prentice Hall. 306 p.

TEMAS IMPORTANTES:

Conceptos de Algoritmos:

Recordar qué es un algoritmo y sus componentes. Además de los libros de la bibliografía, se les recomienda leer los primeros temas del capítulo 1 del libro de:

Neapolitan, R. Foundations of Algorithms. Jones & Bartlett Learning, 5th Edition, 2015.

Pseudocódigo y Diagramas de Flujo:

Las formas de representación básicas de los algoritmos y que han visto desde el inicio de la carrera. Solo recordar los símbolos usados en los diagramas de flujo.

Programas Simples:

Debido a que el examen es de opción múltiple, no se les puede pedir que analicen o desarrollen grandes programas o incluso funciones largas. Por lo que lo más seguro es que vengan programas simples que incluyan los elementos básicos: ciclos, condicionales, arreglos (simples, paralelos, multidimensionales), funciones, parámetros, etc. Es probable que se les pida encontrar el error en un programa o función, indicar qué hace cierto código, indicar el resultado

de alguna función, a partir de una especificación indicar cuál función resuelve el problema dado, etc.

Recursividad y Modularidad:

Una de las maneras de diseño de algoritmos más usada es la de modularidad, así como la técnica de recursividad. Hay que recordar que la modularidad también se estudia como paradigma de programación, pero el tema de paradigmas de programación está más orientado a los lenguajes de programación, que se pregunta en el área 3.2, según la guía. Por lo tanto, en este tema de algoritmos se puede preguntar desde el punto de diseño de algoritmos.

Análisis de Algoritmos:

Arriba se mencionó que se puede preguntar acerca del error en una función, por ejemplo. Esto de alguna manera también es parte del análisis de algoritmos. Sin embargo, el análisis formal implica el verificar el comportamiento de un algoritmo desde el punto de vista del tiempo y del espacio, principalmente. Es decir, cuál es el comportamiento del algoritmo en cuanto el tiempo que tardará en terminar y el espacio en memoria que ocupa. Este tema es extenso, pero al menos hay que conocer lo básico.

Se les recomienda basarse en el libro que se usa en nuestra materia de Algoritmos y Complejidad:

Neapolitan, R. Foundations of Algorithms. Jones & Bartlett Learning, 5th Edition, 2015.

Aunque también el de Cormen (indicado en la bibliografía de la guía) es bueno consultarlo. En cuanto al libro de Neapolitan, con leer el primer capítulo puede que sea suficiente. Estudiar el tema de orden, entender los tipos de funciones (lineal, logarítmica, cuadrática, etc.), los conceptos de big O, theta y omega.

Pueden ver los videos de la materia de Algoritmos y Complejidad:

https://www.youtube.com/watch?v=BAr-EzHYM4I (Importancia de algoritmos eficientes 1/2)
https://www.youtube.com/watch?v=MCxDF8-tp8U (Importancia de algoritmos eficientes 2/2)
https://www.youtube.com/watch?v=4yv38XSOxS0 (Análisis de algoritmos)
https://www.youtube.com/watch?v=49qGo1zaXS4 (Orden 1/2)
https://www.youtube.com/watch?v=Iaf4BQ_Zbjw (Orden 2/2)
https://www.youtube.com/watch?v=9D1ereWiEPY (Análisis de complejidad)

Estrategias Algorítmicas:

Es recomendable recordar algunas de las estrategias algorítmicas más comunes, ya que estas influyen en el diseño de algoritmos. Por lo menos recordar las estrategias de Divide y Vencerás, Programación Dinámica, y Greedy. Esto también lo pueden encontrar en el libro de Neapolitan. Lo importante es tener en mente la idea básica de cada estrategia, ya que podría preguntarse

que cuál estrategia es la mejor para cierto problema, o identificar cuál estrategia está siendo usada en cierto código.

NOTA ACERCA DE LOS LENGUAJES DE PROGRAMACIÓN: Realmente no se especifica el o los lenguajes de programación que serán usados en el examen. En la bibliografía de este tema se tiene el libro de lenguaje C, por lo que se puede asumir que muchas de las preguntas serán en C o con una sintaxis parecida. De cualquier manera, es conveniente familiarizarse con lenguajes como Java, que también es uno de los más comunes.

NOTA ACERCA DE ALGORITMOS "FAMOSOS": Es importante recordar el nombre, lo que hacen y la complejidad, de ciertos algoritmos reconocidos en las ciencias de la computación. Por ejemplo, algoritmos de búsqueda (secuencial, binaria), ordenación (burbuja, inserción, Quicksort, etc.), de manejo de grafos (Floyd, problema del viajero, Dijkstra), matemáticos (serie de Fibonacci, torres de Hanoi, etc.), etc.