

ALGORITMOS Y ESTRUCTURAS DE DATOS 1

¿DE QUE CONSTA EL CURSO?

El curso tiene como objetivo introducir a los estudiantes en los conceptos fundamentales de algoritmia y técnicas avanzadas de programación. Además, busca familiarizarlos con el análisis de algoritmos y su importancia, así como desarrollar habilidades para diseñar soluciones algorítmicas eficientes y resolver problemas de mediana complejidad.

Objetivos específicos:

- Comprender los conceptos básicos de algoritmia y análisis de algoritmos.
- Comprender la importancia de las estructuras de datos a la hora de desarrollar algoritmos.
- Aplicar técnicas de diseño de algoritmos para resolver problemas.
- Desarrollar habilidades prácticas en programación detallada y resolución de problemas.

TEMARIO

Los objetivos antes mencionados se desarrollan en clases teórico-prácticas, dictadas en el salón de clase a través del siguiente temario:

Metodología de resolución de algoritmos.

- Conceptos de algoritmo y estructura de dato.
- Abstracción en programación: particiones-refinamientos.
- Abstracción procedural e introducción a la abstracción de datos.
- Uso de pre y post condiciones.
- Introducción al análisis de algoritmos.
- Eficiencia en espacio de almacenamiento y tiempo de ejecución.
- Evaluación de la eficiencia de un algoritmo.

Array (vectores y matrices)

- Desarrollo teórico, uso y recorridas.
- Utilidad y desarrollo de ejercicios.

Recursión

- Conceptos.
- Programación recursiva: tipos y aplicaciones.

Estructuras dinámicas.

- Estructuras estáticas y estructuras dinámicas.
- Definición e implementación de listas, pilas y colas.

- Algoritmos sobre listas, pilas y colas.

Búsqueda y Ordenación

- Conceptos.
- Algoritmos conocidos.

Tema complementario: Introducción a Java

- Fundamentos del lenguaje.
- Particularidades de Java y buenas prácticas.
- Codificación y ejecución de programas simples.

METODOLOGÍA Y EVALUACIÓN

El curso se desarrolla en **48 horas presenciales**, distribuidas en **3 horas semanales** durante **16 semanas**. La evaluación se basa en:

Se evaluará al alumno con una puntuación de 0 al 100. Esta puntuación se obtiene a través de la sumatoria de:

- Dos evaluaciones escritas y presenciales (parciales), las cuales sumarán 60 puntos entre ambas (30 y 30 respectivamente).
- Tareas domiciliarias obligatorias de carácter periódico (semanal o quincenal), las cuales sumaran 40 puntos en total.

Dichas tareas constaran de:

- Desarrollo de uno o más algoritmos propuestos por el docente.
- Documentación del proceso de desarrollo en un documento con un formato especificado.
- Defensa del trabajo a través de la respuesta a una o más preguntas relacionadas al mismo en la primer media hora de clase del día de entrega.

Aprobación de la materia

- Si el alumno obtiene entre 0 y 69 puntos reprobará el curso.
- Si el alumno obtiene entre 70 y 85 puntos, el alumno obtiene el derecho a rendir un examen en una fecha posterior dictaminada por la institución, el cual es una prueba escrita de 0 a 100 puntos, que, en caso de obtener al menos 70 puntos en ella, aprueba la asignatura.

- Finalmente, si el alumno obtiene una puntuación de 86 a 100 puntos aprueba directamente la asignatura, siendo exonerado de tener que rendir el antes mencionado examen.

MATERIALES , HERRAMIENTAS Y ENTREGAS

El curso se realiza a través de clases teórico – practicas, las mismas son dictadas en el aula por el docente, para llevar a cabo el desarrollo de los prácticos será necesario el uso de una PC, personal o provista por la institución.

Aunque en gran medida la materia es agnóstica a un lenguaje de programación particular, dado que los conceptos que se desarrollan funcionan en casi cualquier plataforma de desarrollo, el lenguaje utilizado será JAVA. En cualquier versión superior a la 8, recomendando utilizar la última disponible. Se recomienda el uso de cualquier tipo de IDE (entorno de desarrollo) que sea compatible con JAVA. Ya sea Netbeans, Eclipse, Visual Studio Code (con los complementos necesarios) o IntelliJ Idea Community Edition, recomendado este último por la claridad de su herramienta de debug.

Los trabajos escritos (parciales) serán 100% en formato escrito a mano.

Las preguntas teóricas referentes a las tareas obligatorias podrán ser escritas a mano, o a través de las herramientas de la plataforma (portal), dependiendo del docente esta decisión.




Las entregas de los algoritmos desarrollados en las tareas obligatorias serán exclusivamente a través del portal, dentro de los tiempos estipulados (fecha y hora).

Formatos y normas de entrega:

Las normas de entrega de las tareas obligatorias deberán ser respetadas por el alumno, en caso de no cumplir con alguna de las directrices, puede conllevar penalidad en la nota del trabajo realizado.

Cada alumno debe entregar un **archivo comprimido (.zip o .rar)** con la siguiente estructura:

NombreApellido_TareaX

```
|—  README.md  
|—  Informe.pdf  
|—  src/  
    |— Main.java  
    |— Algoritmo.java  
    |— Utilidades.java
```

Aclaración: el nombre las clases java dentro de la carpeta **src** son meramente ilustrativos.

Carpeta principal: NombreApellido_TareaX

- **Formato:** ApellidoNombre_TareaX.zip
- **Ejemplo:** GomezCarlos_Tarea1.zip

Archivo README.md

Debe contener:

- **Nombre del estudiante**
- **Materia**
- **Número de la tarea**
- **Descripción breve del algoritmo**
- **Ejemplo de entrada y salida esperada**

Ejemplo de README.md:

****Nombre:**** Carlos Gómez

****Materia:**** Algoritmos y Estructuras de Datos

****Tarea Nro:**** 01

****Tema:**** Ordenamiento

****Descripción:**** Implementación del algoritmo de ordenamiento por selección.

Documentación (Informe.pdf)

Cómo se debe elaborar la documentación se describirá en clase por parte del docente.