

# Identificación de la asignatura

|  |  |
| --- | --- |
| DIVISION ACADEMICA | Ingeniería |
| DEPARTAMENTO | Sistemas |
| PROGRAMA ACADEMICO (Programas académicos a los cuales sirve el curso) | Ingeniería de Sistemas |
| NOMBRE DEL CURSO | Algoritmos y Complejidad. |
| COMPONENTE CURRICULAR AL QUE PERTENECE EL CURSO (Básico, Básico Profesional, Profesional) | Formación básica profesional |
| MATERIA O ÁREA DEL CONOCIMIENTO | Ciencia de la Computación |
| CODIGO DEL CURSO | IST 4310 \_ 01 |
| Número de Registro del Curso (NRC) | NRCs = 3498 / 3499 |
| PRE-REQUISITO (Si lo hay) Especifique código del curso | IST 4031 |
| CO-REQUISITO (Si lo hay)Especifique código del curso |  |
| NUMERO DE CREDITOS SEMESTRALES | 3 |
| TIPO DE CRÉDITO (Obligatorio, Parcialmente libres, libres) | Obligatorio |
| NÚMERO DE SEMANAS | 16 |
| INTENSIDAD HORARIA SEMANAL   1. Número de horas teóricas semanales de trabajo con el profesor 2. Número de horas prácticas semanales de trabajo con el profesor 3. Número de horas semanales de trabajo independiente | 1. 2 2. 2 3. 9 |
| NIVEL DEL CURSO ( Pregrado, Postgrado, Extensión) | Pregrado |
| NOMBRE DEL PROFESOR (PROFESORES) | Ing. José Rafael Capacho |
| UBICACIÓN DEL PROFESOR  (Oficina, e-mail) | Bloque L6, Cuarto Piso,  Oficina No. 4-28.  jcapacho@uninorte.edu.co |
| HORAS DE ATENCIÓN DEL PROFESOR | Lunes a Jueves de 8:30 a 9:45 a.m. |



# Descripción de la asignatura.

**Descripción Sintética de la asignatura**

Complejidad computacional, y su uso en el análisis de algoritmos. Introducción a diferentes estrategias para la solución de problemas mediante la construcción de algoritmos. Análisis general de las clases de complejidad P y NP.

**Descripción ampliada de la asignatura**

Se inicia con una introducción a la noción de complejidad computacional (tiempo y espacio); se explica la complejidad temporal y espacial y su uso en el análisis de algoritmos.

Luego se indican las diferentes estrategias para la solución de problemas mediante la construcción de algoritmos y por último se realiza un análisis general de las clases de complejidad P y NP.

# Justificación.

La asignatura se justifica en términos de:

* Familiarizar al estudiante con las técnicas formales para la obtención de algoritmos eficientes.
* Introducir al estudiante en el análisis formal de problemas intratables determinísticamente.

# Objetivos de la asignatura

**Objetivo General**

El desarrollo del curso pretende que al final el estudiante sea capaz de:

Comprender y aplicar las diferentes técnicas empleadas para el análisis de algoritmos.

**Objetivos Específicos**

Al final del curso se pretende que el estudiante sea capaz de:

1. Conocer uno de los modelos bases para la resolución de problemas del mundo real aplicando la teoría de algoritmos.

2. Conocer y Comprender los conceptos bases de: problemas y algoritmos (sus limitaciones, criterios de mejoramiento y caracterización), con el fin de identificar algoritmos efectivos (eficaces y eficientes).

3. Comprender los conceptos básicos del análisis de complejidad, con base en las notaciones O, Ω y Θ, con el fin de interpretar las clases de complejidad como base para clasificar los problemas tratables o no por ordenador en sus clases P, NP, y NP\_Completos.

4. Aplicar la teoría matemática de ecuaciones de recurrencia en el análisis de la eficiencia en tiempo de algoritmos recursivos.

5. Analizar y diseñar la resolución de problemas aplicando las técnicas de: Dividir y Conquistar, Algoritmos Voraces, Regreso hacia atrás (Backtracking), Programación Dinámica y Teoría de grafos, identificando y programando sus algoritmos básicos útiles en la solución de problemas tratables por ordenador.

# Resultados de Aprendizaje.

Las salidas del curso (Course´s Outcomes (CO´s)) o salidas de aprendizaje que el alumno debe adquirir son las siguientes:

* CO1: Entender cómo caracterizar y clasificar los algoritmos, sus limitaciones y criterios de mejoramiento.
* CO2: Comprender el análisis de complejidad y entender las diferencias de las notaciones O(n), Omega(n), Theta(n).
* CO3: Comprender y aplicar las diferencias de los problemas tipo P, NP y NP\_Completos.
* CO4: Aprender en su análisis, diseño y evaluación las técnicas: Algoritmos Recursivos, Dividir y Conquistar, Algoritmos Voraces, Programación Dinámica, Regreso hacia atrás y Teoría de Grafos.
* CO5: Programar en un lenguaje de programación de alto nivel, tres casos involucrando algunas de la técnicas de: Dividir y Conquistar, Algoritmos Voraces, Programación Dinámica, Regreso hacia atrás, Teoría de grafos, o casos relacionados con los conceptos de problemas P, NP, y NP\_Completos.



# Contenido de la asignatura.

|  |  |  |
| --- | --- | --- |
| **No** | **TOPICO** | **NUMERO DE HORAS** |
| 1 | **MODULO I COMPUTADORES, COMPLEJIDADES E INTRATABILIDAD** | **12** |
| 1.1 | Introducción |  |
| 1.2 | Problemas, Algoritmos , y Complejidad |  |
| 1.3 | Análisis de Complejidad (Complexity Analysis) |  |
| 1.3.1 | Cantidad de trabajo realizado |  |
| 1.3.2 | Análisis asintótico (Asymptotic analysis) |  |
| 1.3.3 | Ordenes de complejidad ( The timing of algorithms ) |  |
| 1.3.4 | Análisis de eficiencia tiempo vs espacio en algoritmos. |  |
| 1.3.5 | Estudio de casos y ejercicios |  |
| 1.4 | Clases de Complejidad (Introducción) (Complexity Classes) |  |
| 1.4.1 | Clases de Complejidad P, NP, NP\_C |  |
| 1.4.2 | Problemas tratables e intratables. |  |
|  |  |  |
| **2** | **MODULO II TECNICAS DE ANALISIS DE ALGORITMOS** | **6** |
| 2.1 | Eficiencia de los algoritmos |  |
| 2.2 | Análisis de programas recursivos |  |
| 2.3 | Resolución de ecuaciones de recurrencia |  |
| 2.3.1 | Ecuaciones de recurrencia lineales |  |
| 2.3.2 | Ecuaciones de recurrencia no lineales |  |
| 2.4 | Ejercicios |  |
| 2.4.1 | Análisis de algoritmos empleando ecuaciones de recurrencias |  |
| 2.4.2 | Comparación de algoritmos iterativos y algoritmos recursivos. |  |
|  |  |  |
| **3** | **MODULO III ESTRATEGIA PARA LA SOLUCION DE PROBLEMAS** | **30** |
| 3.1 | Dividir y Conquistar |  |
| 3.2 | Técnicas heurísticas |  |
| 3.2.1 | Técnicas voraces |  |
| 3.2.2 | Técnicas de acondicionamiento |  |
| 3.3 | Algoritmos de retroceso (Backtracking algoritmos ) |  |
| 3.4 | Programación dinámica |  |
|  |  |  |
| **4** | **MODULO IV EL PROBLEMA DE LA RUTA MAS CORTA** | **6** |
| 4.1 | Algoritmo de Floyd y Warshall |  |
| 4.2 | Algoritmo de Dijsktra |  |
| 4.3 | Equivalencia de problemas |  |
|  |  |  |
| **5** | **MODULO V BUSQUEDA EN TEORIA DE GRAFOS** | **10** |
| 5.1 | Estrategias de búsquedas en grafos |  |
| 5.2 | Métodos de reintento |  |
| 5.3 | Métodos de grafo explícito |  |
|  |  |  |
| **6** | **(\*) MODULO VI COTAS INFERIORES PARA ORDENES DE COMPLEJIDAD** | **-** |
| 6.1 | Algoritmos basados en comparaciones |  |
| 6.2 | Cota inferior : Inserción en una lista ordenada  Máximo de un conjunto, Ordenamiento |  |
|  |  |  |
| **7.** | **(\*) EL PROBLEMA DE EQUIVALENCIA** | **-** |
| 7.1 | El problema de equivalencia |  |
| 7.2 | El algoritmo de Kruskal |  |
| 7.3 | Complejidad computacional amortizada |  |
|  |  |  |
| **8** | **(\*) INTRATABILIDAD** | **-** |
| 8.1 | Introducción |  |
| 8.2 | Máquina de Turing |  |
| 8.3 | Teorema de Cook |  |
| 8.4 | Complejos pero... |  |
| 8.5 | Una alternatica : No-Determinismo. |  |

Nota (\*): Estos tópicos se incluyen en la asignatura a nivel de anexos



* 1. **Evaluación**

|  |  |  |  |
| --- | --- | --- | --- |
| **TIPO** | **PORCENTAJE** | **FECHA** | **TEMA** |
| Primer Parcial (PAR1, I+D) **(\*)** | 20% | 4a Semana | 1,2 |
| Segundo Parcial (PAR2, I+D) **(\*)** | 20% | 8a Semana | 3,4 |
| Examen Final (EF, I+D) | 20% | Según Registro | 1 al 8 |
| Laboratorios (L) | 20% | 3a, 7ª, 9ª, 12a Semana |  |
| I+D (10%) y Quices (10%) (Q) | 20% | 12a Semana |  |

**(\*) Componentes que corresponden al 40%.**

* 1. **Assessment**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Salidas del Curso (SOs) →**  **↓ Competencias adquiridas** | | **CO1** | **CO2** | **CO3** | **CO4** | **CO5** | **Salidas del alumno**  **SOs** |
| Dominio Cognitivo  70% | Conocimiento | Q |  |  |  |  |  |
| Comprensión |  | Q |  |  |  |  |
| Aplicación |  |  | PAR1 |  |  |  |
| Análisis |  | PAR1 |  |  |  |  |
| Diseño |  |  | Q |  |  |  |
| Evaluación |  |  |  | PAR2 | E.FINAL | (6) |
| Laboratorios 20% | |  | L1 | L2 | L3 | L4 | (6) |
| I+D 10% | |  |  | PI | PI | PI |  |

No se tolerará el **plagio** o la **copia**. Sin excepción, en caso de darse uno de estos casos, a los estudiantes involucrados se les iniciará proceso disciplinario, y se actuará conforme al Reglamento de Estudiantes de la Universidad del Norte. El plagio incluye usar contenidos sin la debida referencia, de manera literal o con mínimos cambios que no alteren el espíritu del texto. Adicionalmente, el estudiante debe tener presente que todos los exámenes, informes, trabajos, presentaciones, laboratorios y cualquier otro producto a evaluar debe:

* Ser realizado de forma individual, excepto, en aquellos casos en que el profesor lo autorice por escrito y sin ninguna clase de participación de terceros.
* Debe referenciar en el código fuente las librerías o algoritmos de terceros que hayan sido utilizados. Dar crédito a los autores de las ideas, conceptos teorías, datos en los que se basó el estudiante para realizar su trabajo.
* Debe enviarse por el medio definido por el docente. Si el trabajo debe ser enviado por catálogo web, sólo se recibirá por ese medio, y no por medios alternativos. En caso de no utilizar los medios establecidos para el desarrollo del curso, se asumirá que el trabajo no fue enviado.



# Bibliografía Básica de la asignatura

* 1. **Texto Guía.**
* AHO Alfred V. Ullman Jeffrey D. Foundations of Computer Science C Edition. Computer Science Press. An Imprint of W.H. Freeman and Company. New York. 1.996
* BRASSARD G, y BRATLEY p. Fundamentos de Algoritmia. Prentice Hall Inc., (1997)
* GAREY, Michael R. y JOHNSON, David S. Computers and Intractability : A guide to the theory of NP-Completeness. U.S.A : Bell Laboratories, 1979 . 340 p.
* ROUGHGARDEN, Tim (2020). Algorithms Illuminated. Par 4: Algorithms for NP-Hard Problems. Soundlikeyourself Publishing. LLC. New York.NY.

# Otras referencias.

* AMSBURY Wyne. Data atructures from arrays to priority queues.Wadsworth Publishing Company
* BAASE Van Gelder. Algoritmos Computacionales. Introducción al análisis y diseño. Pearson Education de México S. A. de C. V. (2002).
* BECKMAN, Fank. Mathematical Foundations of Programming. Addison-Wesley, 1980.
* BECERRA S. César. Estructura de Datos en Java 1a. Edición Editorial Kimpres Ltda. Agosto 2000.
* CORMEN Thomas H., LEISERSON Charles E., RIVEST Ronald L. Introduction to Algorithms. Electrical Engineering and Computer Science Departament at the Massachussetts Institute Technology (MIT). Twenty-first printing, 1998.
* GALVE, Javier. GONZALEZ, Juan C. SANCHEZ, Angel y VELAZQUEZ, J. Angel. Algorítmica. Madrid, España: Addison - Wesley Iberoamericana, 1993. 496 p.

1. GREENE, Daniel. Matemathics for the Analysis of Algorithms. Binkhauser, 1982.
2. HARBRON, Thomas. File Systems Structures and Algorithms. Prentice Hall,
3. KNUTH, Donald. El arte de Programar Ordenadores. Reverte, 1987.
4. SEDGEWICK, Robert. Algorithms. U.S.A. : Addison - Wesley Publishng Company, 1995. 711 p.
5. WIRTH, Niklaus. Algoritmos y Estructuras de Datos. Prentice Hall

**Direcciones electrónicas sugeridas**

* <http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-046j-introduction-to-algorithms-sma-5503-fall-2005/video-lectures/>
* <https://www.classcentral.com/course/coursera-algorithms-design-and-analysis-part-1-374>



**Notas importantes**

* La bibliografía dada en ésta parcelación no es de carácter exhaustivo.
* Las evaluaciones, incluidos los laboratorios de programación, se realizarán SIN EXCEPCIÓN en las semanas del semestre en las cuales están programadas, es decir, no se realizaran aplazamientos para ninguna de ellas, debido a que el 80% de la nota definitiva debe estar registrada a más tardar la semana 14.

Actualizado Enero 18 de 2020 ; 10:21 a.m..

ESA

Al SBHU