**ANEXO I**

VIGENCIA DEL PROGRAMA:

|  |
| --- |
| ASIGNATURA: LENGUAJES FORMALES CÓD. DE MATERIA: 07035 |
| CARRERA: LICENCIATURA EN INFORMÁTICA CÓD. DE CARRERA: 288 |

**Ciclo Académico:** Inicial.

Año de la carrera: Segundo.

Horas de clases semanales: 5 horas.

Teóricas: 3 horas.

Prácticas: 2 horas.

Régimen de Cursado: Cuatrimestral / Primer Cuatrimestre.

Observaciones:

**ESPACIOS CURRICULARES CORRELATIVOS PRECEDENTES**

Aprobada/s: Matemáticas Discretas (Cód. 01018) Cursada/s y aprobada/s:

**ESPACIOS CURRICULARES CORRELATIVOS SUBSIGUIENTES**

Asignatura/s: Lenguajes de Programación (Cód. 07054) y Paradigmas de Programación (Cód. 07052)

**1-FUNDAMENTACIÓN**

Esta materia presenta al alumno los basamentos teóricos formales para el entendimiento del funcionamiento de los compiladores y lenguajes de programación. La misma se centra en la tarea de reconocimiento de los elementos del lenguaje y su análisis sintáctico.

Articula con Matemática Discreta, Lenguajes de Programación y Paradigmas de Programación. De la primera materia se necesita todo el conocimiento de grupos y relaciones, así como también, la práctica de manejo de estructuras algebraicas discretas.

Esta materia colabora con las materias subsiguientes presentando los temas de manejos de lenguajes regulares, lenguajes independientes del contexto, jerarquía de lenguajes, basamentos teóricos y prácticos de la construcción de Parsers.

**2- OBJETIVOS GENERALES**

* Introducir al alumno en los fundamentos teóricos de los lenguajes formales.
* Desarrollar la teoría de lenguajes formales a partir de las operaciones con cadenas.
* Enunciar las operaciones entre lenguajes formales en cuanto a conjunto de cadenas.
* Presentar analíticamente las diferencias y relaciones entre gramáticas, lenguajes formales y autómatas.
* Mostrar el valor algorítmico de autómatas finitos y de pila a qué problemas pueden ser aplicados.
* Mostrar analíticamente la relación entre la teoría de las matemáticas, la lingüística y la informática.

**3- OBJETIVOS ESPECÍFICOS**

* Clasificar gramática y lenguajes según las jerarquías de Chomsky.
* Relacionar los conceptos “lenguaje formal”, “gramática” y “autómata”.
* Asociar tipos de autómatas con tipos de gramática.
* Distinguir, por distintos criterios, entre autómatas deterministas y no deterministas.
* Entender distintas implementaciones de autómatas.
* Describir los algoritmos vinculados con los distintos autómatas.
* Formular descripciones teóricas sobre Máquinas de Turing, sus distintas restricciones y la importancia de estas máquinas abstractas en la teoría de la computación.
* Aplicar los conceptos teóricos a la resolución de problemas del mundo real.
* Vincular los conceptos de la materia con problemas vinculados al diseño y desarrollo de software.

**4- CONTENIDO MÍNIMOS**

Lenguajes y gramáticas formales. Clasificación de Chomsky. Lenguajes regulares. Gramáticas regulares. Expresiones regulares. Autómatas finitos. Equivalencias entre gramáticas regulares, expresiones regulares y autómatas finitos. Lema de pumping para lenguajes regulares. Analizadores lexicográficos. Lenguajes independientes de contexto. Gramáticas independientes de contexto. Árboles de derivación. Lema de pumping para lenguajes independientes de contexto. Autómatas de pila. Equivalencias entre gramáticas independientes de contexto y autómatas de pila. Analizadores Sintácticos (Parsers).

**5- PROGRAMA ANALÍTICO**

**Unidad Nº1: Lenguajes Formales**

Definición de Lenguaje Formal, Símbolo, Alfabeto y Palabra.

Palabra: longitud de palabra, palabra vacía, concatenación de dos palabras, potenciación de una palabra, igualdad de palabras, reflexión o reversa de una palabra, prefijo de una palabra, sufijo de una palabra y subpalabra de una palabra.

Símbolo: potenciación de un símbolo.

Lenguaje Formal: diferencias con el lenguaje natural, cardinalidad de un lenguaje formal, lenguajes formales finitos e infinitos, sublenguajes de un lenguaje formal, lenguaje universal, lenguaje sobre un alfabeto, operaciones con lenguajes: unión, intersección, complemento, diferencia, producto, potencia, cierre estrella o de Kleene, cierre positivo y reflexión. Propiedades de los lenguajes.

**Unidad Nº2: Gramáticas Formales**

Definición de Gramática Formal y Producción. Definición formal. Sistemas de reescritura.

Jerarquía de Chomsky. Tipos de gramáticas: tipo 3 o gramática regular, tipo 2 o gramática independiente de contexto, tipo 1 o gramática dependiente de contexto y tipo 0 o gramática irrestricta. Restricciones en las producciones.

Concepto de derivación. Tipos de derivación: derivación en forma horizontal y derivación en forma vertical. Árboles de derivación. Forma sentencial. Sentencia. Lenguaje generado por una gramática. Gramáticas equivalentes. Gramáticas ambiguas y no ambiguas. Recursividad. Niveles de recursividad: producción recursiva, gramática recursiva, recursividad por la izquierda y por la derecha. Factorización a izquierda. Eficiencia en el diseño de gramáticas: limpias y bien formadas. Formas Normales: Forma Normal de Chomsky (CNF), Forma Normal de Greibach (GNF) y Forma Normal de Backus-Naur (BNF).

**Unidad Nº3: Lenguajes Regulares**

Definición de Lenguaje Regular. Ejemplos.

Gramática de un lenguaje regular.

Expresiones regulares. Componentes. Precedencia de operadores. Lenguaje representado por una expresión regular. Propiedades de equivalencias entre expresiones regulares. Expresiones regulares equivalentes y no equivalentes. Expresión Regular Universal. Expresiones regulares y los lenguajes de programación.

Autómata finito. Concepto. Definición formal. Representación gráfica a través de grafos. Representación tabular a través de la Tabla de Transiciones. Reconocimiento de palabras. Diseños incompletos e incorrectos. Autómatas Finitos Incompletos y Completos. Autómatas Finitos equivalentes. Autómatas Finitos Determinísticos. Autómata Finito No Determinísticos. Operaciones con autómatas finitos: complemento, intersección, unión y concatenación.

Algoritmos con Autómatas Finitos: (ER -> AFNλ), (AFNλ -> AFD), (AFD -> AFDmin), (AF -> ER), (GR -> AFD), (AFD -> GR).

Implementación de autómatas finitos.

Lema de pumping para lenguajes regulares.

Aplicaciones.

**Unidad Nº4: Lenguajes Incontextuales**

Definición de Lenguaje Independiente de Contexto.

Gramática de un lenguaje independiente de contexto.

Autómata de Pila. Definición formal. Diseño con transiciones especiales. Autómatas de Pila Deterministas. Autómatas de Pila No Deterministas. Lenguaje aceptado por un Autómata de Pila. Reconocimiento de palabras por estado final y por pila vacía. Análisis de palabras.

Lema de pumping para lenguajes independientes de contexto.

Análisis sintáctico descendente (ASD): Analizadores sintácticos descendentes con retroceso. Analizadores sintácticos descendentes predictivos. Conjuntos de predicción y gramática LL(1). Conversión a gramática LL(1). Tabla de análisis sintáctico LL.

Análisis sintáctico ascendente (ASA): Algoritmo de desplazamiento y reducción. Analizadores sintácticos ascendente con retroceso. Analizadores sintácticos ascendentes predictivos (SLR).

**Unidad Nº5: Máquina de Turing**

Definición de Máquina de Turing. Operación de la máquina de Turing. Diferencias de la máquina de Turing con un autómata finito. Definición formal de máquina de Turing. Configuración de una máquina de Turing. Lenguaje reconocido por una Máquina de Turing. (Turing-aceptable). Aceptación y rechazo de palabras por Máquinas de Turing.

**6- CRITERIOS DE EVALUACIÓN**

Los alumnos deberán conocer los principios de los lenguajes formales regulares e independientes del contexto, las gramáticas que generan esos lenguajes y dominar los conceptos de autómatas necesarios para su identificación; además deben ser capaces de reconocer propiedades y realizar pruebas sobre las mismas a través de demostraciones aritméticas. Comprendiendo las capacidades y limitantes de cada una de estas herramientas y las mejores alternativas para su implementación.

Para demostrar estos conocimientos deben demostrar suficiencia en la resolución de problemas teóricos prácticos, demostraciones e implementaciones en un lenguaje de programación.

Los problemas teóricos prácticos presentados se van correspondiendo con los contenidos dictados en la materia y se trabajan durante la misma, para presentar las técnicas necesarias para resolver dichos problemas.

**7- METODOLOGÍA DE TRABAJO PARA LA MODALIDAD PRESENCIAL**

Las clases serán divididas en contenido teórico y práctico.

Por motivos prácticos ambas clases se dictan en el mismo día y horario; y los segmentos prácticos y teóricos se estructuran en función de los temas dictados.

1. Clases de teoría. Durante las clases teóricas se presentarán los contenidos teóricos, se demostrarán los teoremas y propiedades de cada uno de los conceptos integrados y se resolverán ejemplos similares a los de la práctica que permiten la visualización de los conceptos y técnicas necesarias para la resolución de los mismos.
2. Clases prácticas. En las clases prácticas se realizarán ejercicios similares a los encontrados en la guía de trabajos prácticos y los que serán evaluados en los parciales. Se atenderán todas las dudas de los alumnos y se explicarán todos aquellos temas necesarios para la implementación de los algoritmos y técnicas en un lenguaje de programación. Tienen los objetivos de: i) promover el estudio continúo de los alumnos y ii) de controlar la evolución del aprendizaje de los alumnos. En todas las clases prácticas los alumnos sabrán de antemano qué problemas tienen que realizar. En ellas el profesor dará indicaciones para que los alumnos puedan superar las dificultades que les hayan aparecido en su resolución y resolverá individualmente las dudas y problemas que les hayan aparecido durante su resolución. Será labor del grupo de alumnos elaborar la colección de problemas resueltos de la asignatura.
3. Descripción de las Actividades Prácticas.

Formación Experimental: No se realizan actividades experimentales.

Resolución de Problemas del Mundo Real: Los alumnos deberán resolver Trabajos Prácticos que contienen problemáticas referidas a cada uno de los núcleos temáticos vistos, que corresponden a cada una de las Unidades de los contenidos analíticos. No son de entrega obligatoria.

Actividades de proyecto y diseño de sistemas informáticos: No se realizan.

Instancias supervisadas de formación en la práctica profesional: No se realizan.

Otras actividades: Uso de la herramienta JFlap para modelar cada una de las máquinas abstractas vistas.

1. Simulacros de parcial. La clase previa a cada evaluación parcial se llevará un simulacro que consistirá en plantear a los alumnos situaciones problemáticas similares a las que pueden evaluarse. Instándolos a resolverlos con el apoyo del material elaborado durante las clases, y luego haciendo una puesta en común con el docente y demás alumnos, lo que les permitirá hacer una valoración crítica ante la instancia parcial.

**8- ACREDITACIÓN DE ALUMNOS PRESENCIALES**

Regularización: Los alumnos deberán aprobar dos instancias de examen parcial, teniendo una instancia de recuperación donde pueden recuperar cualquiera de los dos parciales en el caso de haber desaprobado uno. Además, deberán realizar un trabajo práctico de diseño de lenguajes formales. Este trabajo práctico se va trabajando a lo largo de la cursada y el alumno tiene que demostrar suficiencia para resolverlo; el mismo cuenta con una fecha de entrega y pudiéndose recuperarse una vez.

**Promoción: // Pedir texto a Celia o Alfredo**

1. Tener aprobadas las actividades correlativas con examen final al finalizar el cuatrimestre.

Los alumnos deberán aprobar dos instancias de examen parcial con nota 7(siete) o superior, sin recuperar ninguna. Además, deberán realizar un trabajo práctico de diseño de lenguajes formales. Este trabajo práctico se va trabajando a lo largo de la cursada y el alumno tiene que demostrar suficiencia para resolverlo, debe estar aprobado, cumpliendo con su fecha de entrega.

La nota de promoción será el promedio de las notas de los parciales.

**9- METODOLOGÍA DE TRABAJO PARA LOS ALUMNOS EN EL SISTEMA DE ASISTENCIA TÉCNICO PEDAGÓGICO**

Se realizarán a lo largo del cuatrimestre y previo a cada una de las instancias de evaluación, clases de consulta para que los alumnos puedan comprobar su avance.

El seguimiento del trabajo práctico se realiza de forma continua a lo largo del cuatrimestre, presentando soporte y ayuda a su resolución.

Los alumnos cuentan con una guía de ejercicios del mismo tenor de los necesarios para regularizar y aprobar la materia.

**10- ACREDITACIÓN DE ALUMNOS NO PRESENCIALES**

Según Reglamento de Alumnos Vigente.

**11- METODOLOGÍA DE TRABAJO SUGERIDA PARA EL APRENDIZAJE AUTOASISTIDO:** ALUMNOS LIBRES.

Se recomienda la realización de los ejercicios de la guía de trabajos prácticos, así como también los Trabajos prácticos solicitados a los alumnos regulares. La bibliografía además cuenta con ejercitación y demostraciones del mismo tenor que las realizadas en clase. Se recomienda la asistencia al menos a dos clases de consultas previas al final.

**12- ACREDITACIÓN DE ALUMNOS LIBRES**

De acuerdo con el artículo 54 y 55 del Reglamento de Alumnos de la Universidad Nacional del Oeste, y otras reglamentaciones vigentes, se resuelve de la siguiente manera: Se tomará un examen teórico práctico. La parte práctica revisara sobre las ejercitaciones llevadas a cabo en el curso presencial. La parte teórica versará sobre la totalidad del contenido de la materia. La acreditación exige la aprobación de ambas partes.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **13- BIBLIOGRAFÍA** | | | | | | | | |
|  |  |  |  |  |  |  |  |  |
| **Apellido/s Nombre/s** | **Año Edición** | **Título de la Obra** | **Capítulo/ Tomo** | **Lugar de Edición** | **Editorial** | **Biblioteca** | **PMB** | **Otro** |
| **UNO** |
| Hopcroft, J.;  Motwani, R. y  Ullman, J. | 2007 | Teoría de autómatas, lenguajes  y computación. |  | Madrid, España | Addison Wesley |  |  |  |
| Cubero E.;  Moreno M. y  Salomon R. | 2007 | Teoría de autómatas y lenguajes  formales. |  | Madrid, España | McGraw-Hill |  |  |  |
| Kelley Dean | 1995 | Teoría de Autómatas y Lenguajes Formales |  | UK | Prentice Hall |  |  |  |
| Isasi, Martínez y Borrajo | 1997 | Lenguajes, Gramáticas y Autómatas |  | Madrid, España | Addison Wesley |  |  |  |
| Brookshear, J. Glenn | 1993 | Teoría de la Computación. |  | Naucalpa, México | Addison Wesley |  |  |  |
| Aho, Sethi y Ullman | 1986 | Compiladores: Principios, Técnicas y Herramientas |  | Wilmington, EE.UU. | Addison Wesley |  |  |  |

|  |  |  |
| --- | --- | --- |
| **14- VIGENCIA DE PROGRAMA** | | |
| Año | Firma del Profesor Responsable | Aclaración Firma |
|  |  |  |
| Fecha | |

|  |  |  |
| --- | --- | --- |
| **15- VISADO ESCUELA** | | |
| Año | Firma Decano | Aclaración |
|  |  |  |
| Fecha: | |