Ministerio de Cultura y Educación **(Programa del año 2024)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area V: Automatas y Lenguajes

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
FUNDAMENTOS DE COMPUTACION	ING. INFORM.	026/12-08/15	2024	1° cuatrimestre
FUNDAMENTOS DE COMPUTACION	ING. EN COMPUT.	28/12	2024	1° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
APOLLONI, JAVIER MARIANO	Prof. Responsable	P.Adj Exc	40 Hs
LEGUIZAMON, MARIO GUILLERMO	Prof. Colaborador	P.Asoc Exc	40 Hs
LOOR, FABRICIO	Auxiliar de Práctico	A.1ra Simp	10 Hs
VILLEGAS, MARIA PAULA	Auxiliar de Práctico	A.1ra Semi	20 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
Hs	2 Hs	3 Hs	1 Hs	6 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	1º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
11/03/2024	21/06/2024	15	90

IV - Fundamentación

El perfil del ingeniero debe incluir el conocimiento científico necesario que le permita analizar y entender problemas, para los que luego, pueda desarrollar y obtener una solución adecuada, utilizando los beneficios de la tecnología y, en especial, de una computadora. En este contexto, una computadora puede ser vista como un dispositivo de cómputo que realiza tanto operaciones como cálculos de manera automática y que son plasmados en un algoritmo. Al intentar dar una formalización a la idea intuitiva de algoritmo se observa rápidamente que es necesario estudiar los modelos formales de cómputo y con ellos, los lenguajes formales.

Es así que en un primer paso en el estudio de los fundamentos de la computación debe analizarse los lenguajes formales dado la estrecha relación entre estos y los problemas. De los lenguajes formales se desprende la necesidad de estudiar los modelos formales de cómputo asociado a ellos, siendo de especial importancia los modelos basados en dispositivos de reconocimiento o aceptación (autómatas o máquinas de estados) y los modelos basados en dispositivos generadores (gramáticas).

En un segundo paso y a través de dichos modelos formales de cómputo se puede analizar la teoría de la computabilidad. Esta teoría estudia propiedades sobre los lenguajes y permite clasificarlos en decidibles y no decidibles. Con ello se establece un límite entre los problemas efectivamente resoluble (o computables) de aquellos que no pueden ser resueltos (o no

computables) mediante un algoritmo.

Por último, y en un tercer paso en el estudio de los fundamentos de la computación debe analizarse la teoría de la complejidad. En este caso, se estudian las propiedades de los lenguajes decidibles respecto al uso de los recursos, ya sea el tiempo, el espacio o algún otro. La finalidad del estudio es jerarquizar la complejidad de los lenguajes decidibles (y por ende de sus problemas asociados) para dividirlos en lenguajes (o problemas) con soluciones que usen eficientemente el recurso analizado de aquellos que no poseen hasta el momento una solución conocida que utilice eficientemente el recurso analizado.

Este curso aborda todos los elementos teóricos y prácticos necesarios para que la/el estudiante pueda incorporar y comprender los conceptos mencionados en los párrafos anteriores. El abordaje se hará desde una perspectiva que aproxime significativamente a la/al estudiante al aprendizaje incremental y constructivista. Propiciará la integración de los contenidos del curso con los conocimientos adquiridos en cursos previos de la carrera (Matemáticas Discreta, Programación I y II, Estructura de Datos y Algoritmos, etc.), y considerará además el contexto social de la/del estudiante.

V - Objetivos / Resultados de Aprendizaje

Al finalizar el curso se espera que la/el estudiante alcance los siguientes objetivos generales y específicos:

Objetivos generales:
- Vincular los aspectos teóricos y prácticos de la Teoría de Lenguajes, de la Teoría de la Computabilidad y de la Teoría de la Complejidad con la realidad y los problemas de la realidad.
- Fomentar el pensamiento analítico y de interpretación para resolver problemas (de la realidad) dados como enunciados en los ejercicios prácticos utilizando el marco teórico de la materia.
- Mejorar las capacidades y habilidades de la/del estudiante para su futuro laboral.

Objetivos específicos:
-Comprender los conceptos vinculados a la Teoría de los Lenguajes Formales
-Diseñar lenguajes (formales) para representar problemas de la realidad dados en forma de enunciados.
-Comprender conceptos y funcionamientos relacionados con los modelos formales de cómputo: máquinas de estados (autómatas finitos, autómatas push-down y máquina de turing) y gramáticas, y la correspondencia entre ellos y con los lenguajes de programación.
-Diseñar soluciones que resuelvan un problema de la realidad dado en forma de enunciado
utilizando el modelo formal de cómputos más adecuado al problema
- Comprender y analizar los conceptos prácticos y teóricos asociados a la Teoría de la Computabilidad utilizando modelos avanzados de computación como la Máquina de Turing y a partir de ello establecer la decibilidad o no de los lenguajes (lenguajes decidibles y no-decidibles).
- Adquirir la capacidad para discernir cuándo un problema es resoluble y cuándo no lo es analizando propiedades de su lenguaje de decisión asociado
-Comprender y analizar los conceptos vinculados a la Teoría de la Complejidad utilizando modelos avanzados de computación como la Máquina de Turing y con ello establecer la tratabilidad o no de los problema al clasificarlos en P, NP, NP-completo y otros relacionados.
- Adquirir la capacidad para detectar cuándo un problema es tratable y cuándo no lo es analizando propiedades de los lenguajes decidibles.

VI - Contenidos

Unidad 1. Lenguajes Formales
Definición y especificación de los lenguajes formales: alfabeto, cadena, lenguaje. Representación de los lenguajes formales. Dispositivos generadores y reconocedores: Gramáticas y Autómatas. Relación general entre autómata y gramática. Jerarquía de Chomsky.

Unidad 2. Lenguajes Regulares: Definición y Modelos de Cómputo
Lenguajes Regulares (Tipo 3). Expresiones Regulares (ER). Autómata Finito: Autómatas Finitos Determinístico (AFD), Autómatas Finitos No Determinístico (AFND), Autómata Finito No Determinístico con transiciones épsilon. Aplicaciones.

Unidad 3. Lenguajes Regulares: Equivalencias y Propiedades
Equivalencias: AFND a AFD, AFND-epsilon a AFD, ER a AFND-epsilon. Minimización de AFD. Propiedades de clausura de los Lenguajes Regulares.

Unidad 4. Lenguajes Libres del Contexto: Definición y Modelos de Cómputo
Lenguajes Libres del Contexto (Tipo 2). Gramáticas Libres de Contexto. Derivación. Árbol de derivación. Backus Naur Form (BNF) y BNF extendida. Autómata Push Down (autómatas pila): configuraciones, lenguaje aceptado por pila vacía y estado final. Equivalencia entre GLC y APD.

Unidad 5. Lenguajes Recursivamente Enumerables: Modelo de Cómputo
Lenguajes Recursivamente enumerables (Tipo 0). Máquinas de Turing (MT): Máquina de Turing determinística o estándar (MTD), MT como reconocedor y como computadora de funciones numéricas. Extensiones a la MTD: Máquina de Turing No Determinísticas, Máquina de Turing con k-cintas. Tesis de Church-Turing.

Unidad 6. Teoría de la Computabilidad
Computabilidad: Idea intuitiva de Algoritmo. Lenguajes recursivos y recursivamente enumerables: Propiedades. Lenguajes decidibles. definición y ejemplos. Lenguajes no decidibles: Máquina de Turing Universal, reducción de un problema a otro. problema de parada o detención (halting), implicaciones de la no decibilidad de un problema.

Unidad 7. Teoría de la Complejidad Computacional
Complejidad Computacional: Complejidad Temporal y Espacial. Problemas tratables e intratables: jerarquía de complejidades. Clases P y NP. NP-Completitud: Problemas NP-Completos, problema de Satisfactibilidad (SAT). teorema de Cook, reducción, otros ejemplo de problemas NP-Completos estándares.

VII - Plan de Trabajos Prácticos

Durante las semanas que correspondan al dictado se abordarán las 7 unidades del curso tanto desde el contenido teórico como desde las respectivas actividades prácticas. El material de cada unidad estará disponible en el repositorio del curso para que la/el estudiante lo pueda consultar y acceder durante todo el curso.

Las actividades prácticas permitirán el desarrollo y profundización de los contenidos teóricos y será el ámbito donde la/el estudiante alcance los objetivos propuestos para el curso. Las clases prácticas consistirán principalmente en la resolución de ejercicios poniendo en práctica los conocimientos logrados en este curso y aquellos previos utilizando el material didáctico accesible desde el repositorio. En los casos apropiados la/el estudiante podrá hacer uso de las herramientas informáticas que implementan los conceptos teóricos desarrollados en el curso. Se espera que la/el estudiante utilice estas clases prácticas también como ámbito para contrastar ideas y confraternizar con sus compañeros generando y propiciando un espacio de reflexión e intercambio de ideas.

A continuación se desarrolla el plan de trabajos prácticos apuntando en cada caso las habilidades que se esperan que el alumno desarrolle en cada caso.

Plan de Trabajos Prácticos de Aula:

Práctico 1: Alfabetos - Cadenas - Lenguajes.
-Entender el significado de los términos alfabetos, cadenas y utilizarlos para definir lenguajes.
-Operar con cadenas y lenguajes.
-Describir lenguajes mediante una representación finita.
-Vincular los conceptos generales con el de los lenguajes de programación.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal.

Práctico 2: Lenguajes Regulares - Expresiones Regulares - Autómatas Finitos.
-Entender la definición de lenguajes regulares.
-Entender las expresiones regulares (ER) reconociendo operando y operadores.
-Describir lenguajes denotados por las ER.
-Construir ER para que denoten lenguajes específicos.
-Representar problemáticas de la realidad utilizando las ER.
-Analizar y diferenciar las componentes de Autómatas Finitos Determinísticos (AFD).
-Entender el funcionamiento de un AFD.
-Ejecutar un AFD.
-Determinar el lenguaje reconocido por un AFD.
-Diseñar y construir AFD que se ajusten a un lenguaje específico.
-Entender el funcionamiento de un Autómata Finito No Determinístico (AFND) y de un Autómata Finito No Determinístico con transiciones epsilon (AFND-e).
-Ejecutar un AFND y AFND-e.
-Determinar el lenguaje reconocido por un AFND y por un AFND-e.
-Diseñar y construir AFND y AFND-e que se ajusten a un lenguaje específico.
-Discernir las facilidades descriptivas de los 3 modelos de acuerdo al problema o lenguaje específico.
-Representar problemáticas de la realidad utilizando los Autómatas Finitos.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal, y la corroboración de lo realizado manualmente utilizando la herramienta JFLAP o similar.

Práctico 3: Equivalencias entre Autómatas Finitos y Expresiones Regulares. Minimización de Autómatas Finitos.
-Entender y aplicar el algoritmo que obtiene un AFD equivalente a un AFND
-Entender y aplicar el algoritmo que obtiene un AFD equivalente desde un AFND-e
-Entender y aplicar el algoritmo que obtiene un AFND-e que acepta el mismo lenguaje que el denotado por una ER.
-Entender y aplicar el algoritmo que obtiene el AFD mínimo equivalente a un AFD.
-Entender la utilización desde en las problemáticas de la realidad de todos los algoritmos de transformación.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal, y la corroboración de lo realizado manualmente utilizando la herramienta JFLAP o similar.

Práctico 4: Gramáticas Libres del Contexto - Autómatas Push-Down.
-Entender la notación, formas y estructura, ejecución y usos de las Gramáticas Libres del Contexto (GLC).
-Pensar, diseñar y construir GLC que deriven las cadenas de lenguajes específicos.
-Representar problemáticas de la realidad y asociadas a los lenguajes de programación utilizando GLC.
-Entender la notación, formas y estructuras y usos de la notación Backus Nour Form (BNF) para GLC.
-Entender la notación, formas y estructuras y usos de la notación BNF-extendida para GLC.
-Analizar, entender y determinar diferencias y similitudes entre ambas notaciones.
-Analizar, entender y determinar diferencias y similitudes entre ambas notaciones y las GLC.
-Representar problemáticas de la realidad y asociadas a los lenguajes de programación utilizando GLC en notación BNF y BNF-extendida.
-Entender la notación, formas y estructura, ejecución y usos de los Autómatas Push-Down (APD).
-Aprender a discernir entre los Lenguajes Libres del Contexto, aquellos que son inherentemente no determinísticos de los que no los son para poder diseñar el APD adecuado.
-Pensar, diseñar y construir APD Determinísticos y APD No Determinísticos (según sea el caso) que deriven las cadenas de lenguajes específicos.
-Aprender a determinar la conveniencia del uso de uno u otro de los 2 tipos de aceptación en los APD (aceptación por estado final o pila vacía) de acuerdo al lenguaje.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal, y la corroboración de lo realizado manualmente utilizando la herramienta JFLAP o similar.

Práctico 5: Máquina de Turing estándar - Extensiones a la MT,
-Entender la notación, formas y estructura, ejecución y usos de la Máquina de Turing estándar (MTD).
-Pensar, diseñar y construir MTD que reconozcan y acepten las cadenas de lenguajes específicos.
-Pensar, diseñar y construir MTD que computen funciones numéricas y/o realicen procedimientos varios.
-Modelizar problemáticas de la realidad utilizando MTD.
-Entender la notación, formas y estructura, ejecución y usos de la extensión a la MTD: Máquina de Turing con k cintas (MTD-kcintas).
-Pensar, diseñar y construir MTD-kcintas que resuelvan problemas específicos.
-Entender la notación, formas y estructura, ejecución y usos de la extensión a la MTD: Máquina de Turing No Determinística (MTND).
-Pensar, diseñar y construir MTND que resuelvan problemas específicos.
-Aprender a discernir las facilidades descriptivas de los 3 modelos (MTD, MTD-kcintas, MTND) para diseñar y construir el modelo adecuado al problema a resolver.
-Analizar y comparar las similitudes y diferencias del modelo de cómputo más general existente (Máquina de Turing en cualquiera de sus variantes) y de una computadora tanto desde el poder computacional y como el descriptivo (o representacional) para resolver problemas de la realidad.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal, y la corroboración de lo realizado manualmente utilizando la herramienta JFLAP o similar, e implementar una versión simplificada y restringida del modelo de MT en una computadora haciendo uso de un lenguaje de programación.

Práctico 6: Problemas decidibles y no decidibles.
-Entender el vínculo entre problemas y lenguajes formales.
-Analizar, entender y aprender a discernir la diferencia entre lenguajes recursivos y recursivamente enumerables.
-Entender la relación entre lenguajes recursivos y algoritmos.
-Analizar, entender y aprender a diferenciar los problemas decidibles de los problemas no decidibles.
-Pensar y diseñar codificaciones de instancias de los problemas de la realidad y que sean válidas y adecuadas para el modelo de cómputo general de Máquina de Turing.
-Analizar y diseñar soluciones a problemas decidibles utilizando el modelo general de cómputo de la Máquina de Turing.
-Analizar y diseñar soluciones utilizando el modelo general de cómputo de la Máquina de Turing y aplicando el concepto de reducción de un problema a otro para determinar la no decidilidad de ciertos problemas .

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal.

Práctico 7: Complejidad Temporal
-Entender el cálculo de la complejidad temporal en una Máquina de Turing de acuerdo al tamaño de la instancia de entrada (longitud de la cadena de entrada).
-Entender la manera en la que se construye la ecuación para la función de complejidad temporal asociada a una Máquina de Turing M para cadenas de una longitud n (tc_M(n)).
-Analizar una Máquina de Turing M (en cualquiera de sus variantes) y construir la ecuación para la función de complejidad temporal asociada a esa Máquina de Turing M para cadenas de una longitud n (tc_M(n)).
-Analizar la afectación que tiene la variante de Máquina de Turing utilizada para resolver el problema en la complejidad temporal asociada a ese problema.
-Analizar y entender la jerarquía de velocidades de crecimiento de la función de complejidad temporal.
-Analizar y entender las clases de lenguajes o problemas P, NP y NP-Completo.
-Entender el análisis necesario para determinar que un problema es NP-Completo utilizando ejemplos teóricos.
-Analizar y entender el efecto que un problema NP-Completo causa a una ingeniero que está resolviendo un problema de la realidad con esas características.
-Vincular los desarrollos realizados para ejemplos teóricos con los necesarios para ejemplos prácticos de la realidad.

Metodología: Resolver los ejercicios utilizando lápiz y papel y compartiendo ideas con los compañeros de aula en un trabajo grupal.

Plan de Trabajos Prácticos de Laboratorio:

Práctico de Laboratorio:
-Realizar un proceso investigativo mínimo acerca de un problema asociado a alguna de las temáticas de la materia.
-Analizar una posible solución al problema.
-Diseñar e Implementar un algoritmo en un lenguaje de programación que resuelva ese problema.

Metodología: Resolver en forma grupal el práctico utilizando internet, computadora y software adecuado.

VIII - Regimen de Aprobación

La propuesta de trabajo, en lo que respecta al régimen de aprobación, está basada en la evaluación continua o formativa, con el objetivo de relacionar la información sobre la evolución del proceso de aprendizaje de los/las estudiantes con las características de la acción didáctica, a medida que se desarrollan y progresan las actividades de enseñanza y aprendizaje. En este sentido, el estudiante puede regularizar (para luego rendir el examen final) o promocionar. Las condiciones y la metodología propuesta en la que se desarrollará el curso, son las siguientes:

A. Régimen para Estudiantes Regulares

*Aprobar 1 examen parcial práctico o alguna de sus dos respectivas recuperaciones, tal como lo establece la reglamentación vigente.

*Entregar, en tiempo y forma, y aprobar el práctico de laboratorio.

*Entregar, en tiempo y forma, y aprobar el 100% de las evaluaciones periódicas propuestas por la cátedra las cuales consistirán en ejercicios de los prácticos de aula y en caso de ser necesario, ejercicios extras referidos a las temáticas abordadas en los prácticos.

*Como parte del proceso de aprendizaje y con la finalidad de realizar un trabajo colaborativo de comprensión, los docentes efectuarán devoluciones evaluativas de las entregas. Todas las evaluaciones pueden tener una instancia de defensa oral posterior a la entrega donde la/el estudiante exponga, explique y responda dudas o inquietudes de sus compañeros y/o docentes acerca del trabajo realizado en sus entregas.

*Los/las estudiantes deben contar además con un porcentaje mínimo igual o superior al 60% de asistencia y participación activa en las clases prácticas y/o actividades prácticas propuestas por los docentes durante el cursado de la materia.

Nota:
1. Las evaluaciones se aprueban con un porcentaje mínimo de setenta por ciento (70%).
2. El examen parcial se aprueba con un porcentaje mínimo de setenta por ciento (70%) de los ejercicios a resolver. Además se debe desarrollar correctamente al menos el 50% de cada uno de los ejercicios involucrados en el parcial para considerar su aprobación.
3. Si cualquier punto no fuera cumplimentado implica que el estudiantes pase a condición de libre.

B. Régimen para Estudiantes Promocionales
1. Ídem a lo requerido para estudiantes regulares excepto que se debe alcanzar el porcentaje de asistencia mínimo del 80% según lo establecido en la reglamentación vigente.
2. La/El estudiante tendrá que rendir y aprobar un coloquio integrador oral o escrito donde muestre su capacidad de vinculación, integración y análisis de los conceptos teóricos dados durante el transcurso del curso.

**La nota final de promoción se computará promediando la nota promedio de las evaluaciones del punto A y la nota obtenida

en el coloquio integrador del punto B. Cada instancia de evaluación (examen parcial, evaluaciones periódicas, práctico de laboratorio, coloquio integrador de promoción, etc.) se aprueba con una nota igual a 7 o superior, según lo establece la normativa vigente.

C. El curso no admite alumnos en condición de Libre.

D. El examen final puede ser oral y/o escrito.

IX - Bibliografía Básica

[1] Hopcroft J.; Ullman J.; Motwani R. – "Introduction to automata theory, languages and computation". Addison Wesley. 3° Edición. (2007).
[2] Sudkamp, T.A. - "Languages and Machines (An Introduction to the Theory of Computer Science)". Addison Wesley. 3° Edición. (2005).
[3] Sipser, M. – "Introduction to the Theory of Computation". PWS Publishing Company. 2° Edición. (2005).
[4] Wood, D. – "Theory of computation". John Wiley & Sons, Inc. (1988).
[5] Gómez, D. y Pardo, L. M. y Tirnauca, C. - "Lenguajes Formales (para Ingenieros Informáticos)". Universidad de Cantabria (2015). https://personales.unican.es/pardol/Docencia/TALF2012.pdf
[6] Notas de Clase de la Cátedra.

X - Bibliografia Complementaria

[1] Ramón Brena Pinero "Lenguajes Formales y Autómatas" (1997).
[2] Elena Jurado Málaga - "Teoría de autómatas y lenguajes formales". Universidad de Extramadura (2008).
[3] Hopcroft J. y Ullman J.- "Introduction to automata theory, languages and computation". Addison Wesley (1979).

XI - Resumen de Objetivos

Al finalizar el curso se espera que el alumno sea capaz de: (a) comprender la teoría de los lenguajes formales y su jerarquización; (b) discernir las diferencias entre las clases de lenguajes formales; (c) entender el funcionamiento de los modelos formales de cómputo (autómatas finitos, expresiones regulares, gramáticas libres del contexto, autómatas push-down, máquina de turing, autómata linealmente acotado) y sus relaciones con los lenguajes formales; (d) relacionar los lenguajes formales con la estructura y formas de los lenguajes de programación; (e) conocer la relación entre problemas, problemas decidibles y lenguajes formales, lenguajes decidibles; (f) comprender con cierta profundidad la teoría de la computabilidad y sus implicancias en la posibilidad de resolver o no problemas (lenguajes decidibles y no decidibles); y (g) comprender también con cierta profundidad la teoría de la complejidad computacional y a través de ella categorizar a los problemas de acuerdo a su propiedades y dificultades para resolverlos (problemas de la clases P, NP y NP-Completo).

XII - Resumen del Programa

Unidad 1: Alfabetos. Cadenas. Lenguajes. Representación de Lenguajes. Jerarquía de Chomsky.

Unidad 2: Lenguajes Regulares. Expresiones Regulares. Autómatas Finitos.

Unidad 3: Equivalencias. Minimización. Propiedades.

Unidad 4: Gramáticas Libres del Contexto. Autómatas Push-Down. BNF. BNF-extendida

Unidad 5: Máquinas de Turing. Extensiones a la Máquinas de Turing.

Unidad 6: Teoría de la Computabilidad. Relación entre problemas y lenguajes. Lenguajes Recursivos y Recursivamente Enumerables. Problemas decidibles y no decidibles.

Unidad 7: Teoría de la Complejidad Computacional. Clases de complejidad temporal y espacial. Problemas P y NP. Problema NP-completos.

XIII - Imprevistos

Correo de contacto: javierma@email.unsl.edu.ar (Javier Apolloni)

XIV - Otros