1. Objetivos

- Transmitir a los alumnos conceptos fundamentales vinculados al diseño de lenguajes de programación para facilitar el análisis de los distintos paradigmas.
- Hacer operativos los conceptos básicos, mediante procesos de reflexión, creación y ejercitación.
- Desarrollar conocimientos y competencias vinculadas con el diseño y criterios de diseño de lenguajes de programación.
- Desarrollar competencias para la identificación y selección de paradigmas.
- Integrar los conocimientos particulares, vinculados al diseño de lenguajes de programación, con otros conocimientos pertinentes, ya logrados durante el curso de la carrera.
- Ajustar la visión individual conformada desde la teoría y la práctica realizada durante la carrera sobre la programación, los lenguajes de programación y su implementación a su marco teórico actual; completarla.

Al finalizar el curso los alumnos estarán preparados para

- Analizar y juzgar críticamente cualquier lenguaje de programación en función de su aplicabilidad, estructuras, componentes e implementación.
- Clasificar lenguajes de programación en función de su uso, aplicabilidad e implementación.
- Vincular las características de los lenguajes de programación con los criterios que determinaron su diseño (epistemología: conocimientos teóricos, sintaxis, semántica) y a las posibilidades de implementación (estado del arte: hardware, software).
- Vincular los distintos paradigmas con su modelo semántico.
- Aplicar los contenidos de la asignaturas a la resolución de problemas del mundo real.
- Transferir los conocimientos adquiridos en el contexto teórico a otras situaciones de la práctica.

2. Contenidos

Unidad 1
Evolución de los principales lenguajes de programación. Perspectiva epistemológica / histórica de los principales lenguajes de programación. Criterios de implementación. Lenguajes iniciales: FORTRAN y COBOL. Lenguajes de la década del 60: Algol 60. Influencia del Algol en los lenguajes superiores. Lenguajes de la década del 70: Pascal, C. Lenguajes orientados a objetos: C++. Lenguajes visuales. Perspectiva epistemológica: concepto de paradigma, nociones básicas de semántica formal. Paradigmas de lenguajes de programación.

Unidad 2
Gramáticas independientes del contexto. Gramáticas regulares. Sintaxis de los lenguajes de programación. Reglas sintácticas. Notaciones. Reglas de producción. Derivaciones. Árbol de Parsing. Árboles de análisis sintáctico. Fases de compilación de un programa.

Unidad 3
Semántica de los lenguajes de programación. Nociones básicas de semántica formal. Análisis léxico y sintáctico. Traducción. Concepto de máquina virtual.
Relación entre las jerarquías de Chomsky y la evolución de los lenguajes de programación. Paradigma imperativo. Assembler: semántica y sintaxis.

Unidad 4
Entidades y ligaduras Noción de ligadura. Ligadura estática y ligadura dinámico. Variables.. Vida y visibilidad de variables. Expresiones y sentencias de asignación.

Unidad 5
Subprogramas y su implementación. Registro de activación. Llamado a subrutinas (FORTRAN).Administración de registros de activación en Algol. Variables dinámicas. Arreglos flexibles. Organización de los registros de activación. Cadenas estáticas y dinámicas. Pasajes de parámetros.

Unidad 6
Programación estructurada. Estructuras de control. Teorema de Bohm y Jacobbini. Programación modular. Abstracción y encapsulamiento. Módulos, abstracciones en el diseño, abstracciones de control, abstracciones de datos. Programación orientada a objetos. Abstracciones. Implementaciones.

Unidad 7
Sistemas de tipos. Tipos de datos. Tipos construidos en el lenguaje. Agregados de datos: Producto cartesiano, arreglos, secuencias, conjuntos. Tipos de datos abstractos. Verificación de tipos y ámbitos. Verificación fuerte de tipos. Conversiones entre tipos. Conversiones implícitas y explícitas. Polimorfismos, niveles de polimorfismos.

Unidad 8
Paradigma funcional. Cálculo lambda. Lenguaje LISP, semántica y sintaxis. Análisis de LISP en función de los conceptos *ligadura* y *registro de activación*. Implementación. Aplicaciones LISP.

Unidad 9
Paradigma lógico. Cláusulas de Horn. Lenguaje PROLOG, semántica y sintaxis. Análisis de PROLOG en función de los conceptos de la teoría de los lenguajes de programación.

3 Bibliografía:

Básica

WATT, D. :Programming Languages Concepts and Paradigms", Hoare Series Editor, Prentice Hall, (1990) http://www.acm.org

LOUDEN, K. C. Lenguajes de programación: Principios y prácticas. Colección Ciencias Ingeniería. Editorial Cengage Learning / Thomson Internacional. ISBN 9789706862846 (2004)

Bibliografía complementaria:

BACKUS, J. Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs. In: Comm. of the ACM, Vol. 21, Nro. 8, pp. 613-641 (1978). http://www.stanford.edu/class/cs242/readings/backus.pdf

CARDELLI, L. and WEGNER, P. On Understanding Types, Data Abstraction, and Polymorphism. In: Computing Surveys, Vol 17 n. 4, pp 471-522, (1985) http://lucacardelli.name/Papers/OnUnderstanding.A4.pdf

COLMERAUER, A. y ROUSSEL, P.: The birth of Prolog (1992) http://alain.colmerauer.free.fr/ArchivesPublications/HistoireProlog/19november92.pdf

KNUTH, D. E. von Neumann's First Computer Program. In: Computing Surveys, Vol. 2, Nro. 4, pp: 247-260 (1970) http://www.acm.org

PARNAS, D.L. On the Criteria To Be Used in Decomposing Systems into Modules. In: Comm. of the ACM, Vol. 15, Num. 12 , pp: 1053-1058 (1972) http://www.cs.umd.edu/class/spring2003/cmsc838p/Design/criteria.pdf
WATT, D. :Programming Languages Concepts and Paradigms", Hoare Series Editor, Prentice Hall, (1990) http://www.acm.org

Herramientas de trabajo :

- Emulador de Assembler
- Compilador Pascal o C

- Intérprete Lisp
- Intérprete Prolog
- Entorno de trabajo Java

4. Metodología de enseñanza

La metodología de trabajo de esta materia se corresponde a la formulada en KRONE, Joan (2006) *A Project Approach to the Theory of Programming Languages* en CCSC: Midwestern Conference, JCSC 22, 1 (October 2006), (pp. 84-93).

En este trabajo, la autora describe las características propias de esta materia y fundamenta el aprendizaje a través del desarrollo de proyectos sobre diferentes temas que pueden ser llevados a cabo individualmente o en equipos. En el mismo trabajo se enuncian los tipos de proyectos que se podrían desarrollar.

A continuación se describe la forma en que se implementa dicha metodología en este curso:

1. Los alumnos son informados con respecto a los contenidos de la materia, sus características, su importancia en el plan de estudios, en su formación y en su desempeño profesional.
2. A partir de las características de los contenidos que deberán ser tratados, se los pone al tanto de la metodología que se llevará a cabo:
    a. Selección de un tema de una lista de trabajos posibles.
    b. Interacción con los alumnos con respecto a la visión que ellos tiene de la resolución del posible trabajo.
    c. Los alumnos definen el trabajo que realizarán en función de los temas ofrecidos, los contenidos del programa y su propia motivación.
3. Los temas seleccionados por los alumnos son tenidos en cuenta por el docente durante la presentación teórica, de modo que los alumnos puedan capitalizar la teoría para y en el desarrollo de sus trabajos.
4. Contenidos del trabajo:
    a. Desarrollo teórico fundamentado, citas formuladas adecuadamente.
    b. Ejemplos prácticos de las situaciones y casos descriptos en el desarrollo teórico.
5. Formato del trabajo:
    a. Título
    b. Abstract / resumen
    c. Palabras claves
    d. Introducción
    e. Desarrollo
    f. Conclusiones
    g. Bibliografía.

Además de este trabajo, que se desarrollará a lo largo del curso de la materia y que debe tener valor integrador, se realizarán prácticas sobre computadora para experimentar las características de la programación en los distintos paradigmas.

5. Evaluación

5.1 Oportunidades y criterios de Evaluación

a. En cada clase.
Mediante actividad práctica realizada en computadoras, se vincularán los contenidos introducidos en la teórica en los trabajos prácticos que desarrollan los alumnos. Los alumnos deberán entregar un informe parcial de dicho trabajo, que involucre la práctica desarrollada, al comienzo de la clase siguiente.

Se considerará:
Durante las sección teórica.

- Calidad de la participación durante las clases teóricas:
- ¿Integra los conocimientos propios de la materia con otras materias de la carrera?

Durante la sección práctica

- Construcción de criterios para la toma de decisiones
- ¿Se Incluye los conocimientos tratados durante la sección teórica?

En la resolución de trabajos prácticos (resolución de problemas de la disciplina)

- Trabajos prácticos completos:
- El alumno presenta la carpeta de trabajos prácticos.

b. Examen parcial
Los alumnos serán evaluados con respecto a los contenidos presentados en clase mediante a) preguntas sobre contenidos teóricos y la resolución de ejercicios y b) sobre la autoría y dominio del trabajo que están desarrollando mediante preguntas orientadas a relacionar los contenidos de la materia con cada trabajo en particular; también se indagará sobre la participación del alumno en la producción de dicho trabajo.

En evaluación parcial

Se considerará:

- Especificidad en las respuestas a las preguntas en evaluaciones parciales.
- Las respuestas son concretas y claras.

c. Desarrollo del trabajo práctico integrador
Los alumnos acompañarán el desarrollo de la teoría de la materia mediante la resolución de un problema abierto, que involucra situaciones de diseño y programación, posible en la actividad profesional, ajustado al perfil del Licenciado en Sistemas de Información.
La extensión y profundidad del tratamiento del tema deberá ser adecuada al año de la carrera en que esté ubicada la asignatura.

En la presentación del trabajo práctico final (actividad de proyecto y diseño)

Se considerará:

- Especificidad en la formulación y calidad de la presentación final del trabajo de la materia.
- El trabajo final de la materia cumple con las especificaciones y se fundamenta en los conceptos que fueron desarrollados durante las clases teóricas.
- El trabajo incluye documentación del programa generado.

d. Examen final
El examen final se desarrollará a partir del trabajo que el alumno produjo a lo largo de toda la materia. A partir del mismo se indagará sobre el dominio que tiene el alumno de los conceptos propios de la materia y su capacidad para ponerlos en juego.

5.2 Requisitos para la aprobación

Aprobación de la cursación de la asignatura. Para aprobar es necesario cumplir con:

- Asistencia mínima del 50%

- Aprobación del examen parcial con nota igual o superior a cuatro puntos:

  Los parciales deben rendirse en las fechas estipuladas por la Facultad.

  En el caso de que el alumno desapruebe el parcial cuenta con una instancia de recuperación.

  El desaprobar o no asistir a la recuperación (teniendo el parcial desaprobado) tiene como consecuencia desaprobar el curso de la materia.

- Aprobación de los Trabajos prácticos con nota igual o superior a cuatro puntos:

  En el caso de esta materia la nota final de los trabajos prácticos se calcula como una nota promedio de los trabajos requeridos. Existe una instancia de recuperación.

Aprobación de la asignatura. Para aprobar es necesario aprobar la cursación y el examen final

- Para aquellos alumnos que no alcanzaran el 75% de asistencias el examen final escrito se aprueba con seis puntos.
- Para los alumnos que alcancen o superen el 75% el examen final se aprueba con cuatro puntos.