FORMULARIO PARA LA PRESENTACIÓN DE PROGRAMAS DE ASIGNATURAS

Año Lectivo: 2023

UNIVERSIDAD NACIONAL DE RÍO CUARTO

FACULTAD DE CIENCIAS EXACTAS FISICO-QUIMICAS Y NATURALES

DEPARTAMENTO DE COMPUTACION

CARRERA/S: Licenciatura en Ciencias de la Computación

PLAN DE ESTUDIOS: Año 1999 – Versión 1

ASIGNATURA: Taller de Diseño de Software CÓDIGO: 3306

MODALIDAD DE CURSADO: Presencial

DOCENTE RESPONSABLE: Francisco Bavera, Doctor en Ciencias de la Computación, Profesor Adjunto Exclusivo

EQUIPO DOCENTE: Cecilia Kilmurray, Doctora en Ciencias de la Computación, Ayudante de Primera Simple

RÉGIMEN DE LA ASIGNATURA: cuatrimestral

UBICACIÓN EN EL PLAN DE ESTUDIO: cuarto año, segundo cuatrimestre

RÉGIMEN DE CORRELATIVIDADES:

Asignaturas aprobadas: Análisis y Diseño de Sistemas (3303), Base de Datos (1959)

Asignaturas regulares: Ingeniería de Software (3304)

CARÁCTER DE LA ASIGNATURA: Obligatoria

CARGA HORARIA TOTAL: 210 horas

Teóricas:	**28 hs**	**Prácticas:**	**28 hs**	**Teóricas - Prácticas:**	**- hs**	**Laboratorio:**	**154 hs**

CARGA HORARIA SEMANAL: 15 horas (según el plan de estudio vigente)

Teóricas:	**2 hs**	**Prácticas:**	**2 hs**	**Teóricas - Prácticas:**	**- hs**	**Laboratorio:**	**11 hs**

1. CONTEXTUALIZACIÓN DE LA ASIGNATURA

Los temas involucrados en el diseño e implementación de Compiladores conforman una parte muy relevante en la currícula de Ciencias de la Computación. El desarrollo de un compilador involucra gran cantidad de técnicas y métodos que permite aplicar una cantidad considerable de conceptos teóricos y de implementar técnicas y algoritmos en casos específicos.

En la materia Taller de Diseño de Software, se aplican un número considerable de técnicas, muchas de estas técnicas son casos particulares de técnicas fundamentales. Pero, se deja librado al alumno: (1) asociar la aplicación de determinada técnica en otros contextos; (2) determinar, o no, la generalización de la técnica; y (3) investigar, o no, otras alternativas de diseño e implementación.

Se guía al alumno para que: (1) pueda vislumbrar que el campo de aplicación de las técnicas dadas es más amplio; y (2) reconozca que existen una gran cantidad de técnicas y algoritmos, no todas vistas en clase, para diseñar e implementar un compilador.

Los contenidos fueron seleccionados con el fin de que el alumno cuente con los conocimientos necesarios para culminar exitosamente el proyecto. Las actividades se seleccionaron con el fin de cumplir con los objetivos propuestos, los cuales, todos contribuyen a fortalecer el perfil del egresado y su práctica profesional.

Se hace especial hincapié en desarrollar la autonomía del alumno para aprender y utilizar las herramientas involucradas en el desarrollo del proyecto (flex, yacc, bash, C, Makefile).

Conocimientos esperados de los alumnos de las materias correlativas: métodos y herramientas de diseño y análisis de sistemas. Métodos y prácticas de testing. Documentación.

Otros conocimientos necesarios de materias previas:
Algoritmos I: Estructuras de datos. Implementación de listas (simple y doblemente encadenadas), pilas, diccionarios y tablas de Hashing.
Análisis Comparativo de Lenguajes: conocimientos básicos del Lenguaje C, manejo de memoria dinámica, pasaje de parámetros y frames de ejecución, conceptos de: compiladores, interpretes, ensambladores y linkers.
Organización del Procesador: Lenguajes ensambladores.

2. OBJETIVOS PROPUESTOS

3. Capacidad de aplicar los conocimientos adquiridos hasta el momento a la construcción de soluciones computacionales concretas, autodocumentadas, eficaces y eficientes.
4. Capacidad autónoma de adquirir destreza en el uso de nuevos herramientas de desarrollo de software.
5. Habilidad en el manejo del ambiente de desarrollo Linux.
6. Habilidad en el uso del lenguaje C.
7. Capacidad de abordar el diseño de sistemas complejos.
8. Capacidad de realizar la descomposición modular de un proyecto y de trabajar en grupo.
9. Habilidad en el uso de herramientas para la construcción de compiladores y otros

procesadores de lenguajes.

10. Habilidad para realizar el *testing* modular y global de un sistema complejo
11. Capacidad de redactar documentación técnica.
12. Capacidad comprender y utilizar bibliografía escrita en inglés.

13. EJES TEMÁTICOS ESTRUCTURANTES DE LA ASIGNATURA Y ESPECIFICACIÓN DE CONTENIDOS

3.1. Contenidos mínimos (según plan de estudio vigente)

Mediante el desarrollo de un proyecto que dé solución a un problema real, se deberá integrar los conocimientos adquiridos en las asignaturas cursadas. Se deberán desarrollar todas las etapas y producir toda la documentación referente al proyecto, de acuerdo a los estándares en uso.

3.2. Ejes temáticos o unidades

Unidad 0. Introducción. Compiladores e interpretes. Fases del compilador. Diseño de un compilador. Ensambladores. Enlazadores.

Unidad I. Gramáticas de atributos. Atributos sintetizados y heredados. Árbol decorado de una sentencia, evaluación, grafo de dependencia, gramáticas evaluables. Definiciones guiadas por sintaxis. Esquemas de traducción. Construcción de árboles sintácticos.

Unidad II. Aplicación de las Definiciones guiadas por sintaxis a la verificación de tipos y la generación de código. Uso de herramientas lex y yacc.

Unidad III. Código Intermedio. Ventajas de introducir una fase de código intermedio. Lenguajes de código intermedio: Usando Grafos y Árboles Sintácticos. Máquinas Abstractas, Máquinas Pila y máquina de tres direcciones. Generación de código para las distintas representaciones.

Unidad IV. Generación de código objeto. Criterios de optimización. Bloques básicos de un programa. Determinación de bloque básicos. Optimizaciones avanzadas. Análisis estático de código. Generación de código objeto en lenguaje ensamblador.

14. ACTIVIDADES A DESARROLLAR

La asignatura está organizada como un taller o laboratorio, en el cual, mediante la realización de un proyecto se revisan y utilizan los conceptos estudiados en otras asignaturas y más específicamente en Autómatas y Lenguajes, Algoritmos I y Análisis Comparativo de Lenguajes.

El proyecto consiste en diseñar y desarrollar un compilador para un subconjunto de un lenguaje de programación. El proyecto se divide en 6 etapas que abarcan todo el proceso de compilación: análisis léxico, análisis sintáctico, análisis semántico, generación de código intermedio, generación de código objeto (assembly) y optimización del código. Cada una de las etapas consta de diseño, implementación y testing de los artefactos desarrollados. Los contenidos básicos a desarrollar son aquellos relacionados con cada etapa: análisis léxico, análisis sintáctico, análisis semántico, código intermedio, código objeto, generación de código intermedio y objeto y optimización del código.

CLASES TEÓRICAS:

Teórico Nº1: Introducción
Teórico Nº2: Análisis Léxico

Teórico Nº3: Análisis Sintáctico
Teórico Nº4: Análisis Semántico
Teórico Nº5: Traducción Dirigida por la Sintaxis
Teórico Nº6: Traducción Dirigida por la Sintaxis (II)
Teórico Nº7: Generación de Código Intermedio
Teórico Nº8: Generación de Código Objeto
Teórico Nº9: Generación de Código Objeto (II)
Teórico Nº10: Análisis de Código y Optimizaciones
Teórico Nº11: Análisis de Código y Optimizaciones (II)

CLASES PRÁCTICAS:

Trabajo Práctico Nº1: Análisis Léxico y Sintáctico
Trabajo Práctico Nº2: Análisis Semántico
Trabajo Práctico Nº3: Traducción Dirigida por la Sintaxis
Trabajo Práctico Nº4: Generación de Código Intermedio
Trabajo Práctico Nº5: Generación de Código Objeto

CLASES DE TRABAJOS PRÁCTICOS DE LABORATORIO:

Etapa Proyecto Nº1: Análisis Léxico y Sintáctico
Etapa Proyecto Nº2: Análisis Semántico
Etapa Proyecto Nº3: Traducción Dirigida por la Sintaxis
Etapa Proyecto Nº4: Generación de Código Intermedio
Etapa Proyecto Nº5: Generación de Código Objeto
Etapa Proyecto Nº6: Optimizaciones

Implementación del proyecto en modalidad grupal. Carga horaria semanal 11 hs., 4 hs. sincrónicas de discusión, diseño y presentación de lo realizado, 7 hs. de trabajo colaborativo entre los integrantes de cada grupo.

15. PROGRAMAS Y/O PROYECTOS PEDAGÓGICOS INNOVADORES E INCLUSIVOS

16. CRONOGRAMA TENTATIVO DE CLASES E INSTANCIAS EVALUATIVAS

Semana	Día/Horas	Actividad: tipo y descripción*
1	25/8 – 3 hs.	Teórico: Introducción – Presentación del Proyecto
2	30/8 – 3 hs.	Teórico-Práctico: Análisis Léxico
2	1/9 – 2 hs.	Laboratorio - Consulta
3	6/9 – 3 hs.	Teórico-Práctico: Análisis Sentáctico
3	8/9 – 2 hs.	Laboratorio – Consulta - Evaluación: Entrega Primer Etapa
4	13/9 – 3 hs.	Teórico-Práctico: Análisis Semántico
4	15/9 – 2 hs.	Laboratorio - Consulta
5	22/9 – 3 hs.	Teórico-Práctico: traducción dirigida por la sintáxis
5	24/9 – 2 hs.	Laboratorio – Consulta - Evaluación: Entrega Segunda Etapa
6	27/9 – 3 hs.	Teórico-Práctico: traducción dirigida por la sintáxis
6	29/9 – 2 hs.	Laboratorio - Consulta

7	4/10 – 3 hs.	Teórico-Práctico: Generación código intermedio
7	6/10 – 2 hs.	Laboratorio - Consulta
8	11/10 – 3 hs.	Teórico-Práctico: Generación código objeto
8	13/10 – 2 hs.	Laboratorio – Consulta - Evaluación: Entrega Tercer Etapa
9	18/10 – 3 hs.	Teórico-Práctico: Generación código objeto
9	20/10 – 2 hs.	Laboratorio - Consulta
10	25/10 – 3 hs.	Teórico-Práctico: Análisis de Código - Evaluación: Entrega Cuarta Etapa
10	27/10 – 2 hs.	Laboratorio - Consulta
11	1/11 – 3 hs.	Teórico-Práctico: Optimizaciones
11	3/11 – 2 hs.	Laboratorio - Consulta
12	8/11 – 2 hs.	Laboratorio - Consulta
12	12/11 – 2 hs.	Evaluación: Entrega Quinta Etapa
13	15/11 – 2 hs.	Laboratorio - Consulta
13	17/11 – 2 hs.	Laboratorio - Consulta
14	22/11 – 2 hs.	Laboratorio - Consulta
14	24/11 – 2 hs.	Evaluación: Entrega Final – Defensa del Proyecto

*Teóricos, teóricos-prácticos, trabajos de laboratorios, salidas a campo, seminarios, talleres, coloquios, instancias evaluativas, consultas grupales y/o individuales, otras.

17. BIBLIOGRAFÍA

7.1. Bibliografía obligatoria y de consulta

De lectura obligatoria:

- "Compilers: Principles, Techniques, and Tools" Second Edition, by Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman and Monica S. Lam. 2006.
- Modern Compiler implementation in Java. Andrew Appel, Cambridge University Press, 1998.
- Documentación (on line) de las herramientas del sistema operativo usadas *gnu assembly language, Java, gnu debbuger, JFlex, CUP*.

De consulta:

- Advanced Compiler Design and Implementation. Steven Muchnick. Morgan Kaufmann, 1997.
- Modern Compiler Design. Grune, Bal, Jacobs,Langendoen. John Wiley & Sons. 2001.
- Assembly Language and Programming, Peter Abel. Prentice Hall, 1995
- " A retargetable C Compiler. Fraser & Hamson", Cumming 1995
- "An introduction to Compiler Constructions". Waite, Carter . Harper Collins 1992

7.2. Otros: materiales audiovisuales, enlaces, otros.

ClassRoom como aula virtual (repositorio, foros, etc), Slack para al comunicación,

18. DÍA Y HORARIOS DE CLASES

Lunes y miercoles de 8 a 12 hs.

19. DÍA Y HORARIO DE CLASES DE CONSULTAS

A coordinar con los alumnos.

20. REQUISITOS PARA OBTENER LA REGULARIDAD Y LA PROMOCIÓN

- **CONDICIONES DE REGULARIDAD:** Aprobación de las entregas parciales y culminación de la implementación del proyecto.

- **CONDICIONES DE PROMOCIÓN:** Aprobación con nota mayor a 7 del proyecto, entregas puntuales de cada etapa, culminación de la implementación del proyecto, presentación del informe del proyecto, exposición y defensa del mismo.

21. CARACTERÍSTICAS, MODALIDAD Y CRITERIOS DE LAS INSTANCIAS EVALUATIVAS

La forma de evaluación consta de tres partes: (1) seguimiento del desarrollo del proyecto (puntualidad en las entregas, desempeño, participación, calidad, corrección, claridad, metodologías usadas y justificación de las actividades realizadas); (2) evaluación, siguiendo los mismos lineamientos, en la entrega final; y (3) evaluación en el desempeño de los alumnos en la resolución de problemas.

- **Evaluaciones Parciales:** Entrega de cada una de las etapas del proyecto. En cada etapa se fundamentalmente la capacidad de resolución de problemas y de implementación de soluciones utilizando los contenidos introducidos. Como las etapas son incrementales en cada etapa se deben incluir las correcciones de la etapa anterior.

- **Evaluación Final:** Exposición y defensa del proyecto y un examen oral. Se evalúa fundamentalmente la adquisición de los conceptos fundamentales, su vinculación con el resto de la carrera y la capacidad de aplicarlos.

- **Examen Regular:** Presentación del informe, defensa del proyecto y evaluación sobre los conceptos fundamentales involucrados, su vinculación con el resto de la carrera y la capacidad de aplicarlos.

- **Examen Libre:** entrega de todas las etapas de un proyecto y el informe final (con una antelación de al menos 72 hs. al examen). Defensa del proyecto y evaluación sobre los conceptos fundamentales involucrados, su vinculación con el resto de la carrera y la capacidad de aplicarlos.

Firma Profesor/a Responsable **Firma Secretario/a Académico/a**