FORMULARIO PARA LA PRESENTACIÓN DE PROGRAMAS DE ASIGNATURAS

Año Lectivo: 2024

UNIVERSIDAD NACIONAL DE RÍO CUARTO
FACULTAD DE CIENCIAS EXACTAS, FÍSICO-QUÍMICAS Y NATURALES
DEPARTAMENTO DE COMPUTACIÓN

CARRERA/S: Licenciatura en Ciencias de la Computación y Analista en Computación

PLAN DE ESTUDIOS: 1999

ASIGNATURA: Programación Avanzada **CÓDIGO:** 1948

MODALIDAD DE CURSADO: Presencial

DOCENTE RESPONSABLE: Dr. Pablo Castro

EQUIPO DOCENTE: Lic. Agustín Borda Ay. 1ra Simple, Prof. E. Cerdá, Ay. de Primera semi, Lic. Nicolas Streri, Ay. de primera semi.

RÉGIMEN DE LA ASIGNATURA: Cuatrimestral

UBICACIÓN EN EL PLAN DE ESTUDIO: Primer Cuatrimestre, Segundo Año

RÉGIMEN DE CORRELATIVIDADES:

Asignaturas aprobadas: Lógica Matemática Elemental

Asignaturas regulares: Introducción a la Programación y Algorítmica

CARÁCTER DE LA ASIGNATURA: Obligatoria

CARGA HORARIA TOTAL: 168 horas

Teóricas:	**56 hs**	**Prácticas:**	**56 hs**	**Teóricas - Prácticas:**	**…. hs**	**Laboratorio:**	**56 hs**

CARGA HORARIA SEMANAL: 8 horas

Teóricas:	**4 hs**	**Prácticas:**	**2 hs**	**Teóricas - Prácticas:**	**…. hs**	**Laboratorio:**	**2 hs**

1. CONTEXTUALIZACIÓN DE LA ASIGNATURA

La materia provee a los estudiantes con conceptos de programación avanzados tales como el desarrollo de programas correctos tanto en programación funcional como en programación imperativa. Así como también conceptos de lenguajes y automatas y básicos de computabilidad. Estas nociones son necesarias para las materias de tercer año que complementan estos contenidos con técnicas de ingeniería de software y paradigmas de programación. La materia tiene un uso importante de lógica, sobre todo para el razonamiento sobre programas, así como de nociones de programación básicas. Estos conceptos proveen la teoría necesaria para que los graduados pueden desarrollarse en el ámbito profesional con metodologías adecuadas para el desarrollo de software, así como les permite comprender conceptos fundamentales de calidad de software.

2. OBJETIVOS PROPUESTOS

Que los alumnos sean capaces de:

• Desarrollar habilidades para el desarrollo formal de programas.

• Comparar críticamente los paradigmas de programación funcional e imperativo, desde el punto de vista de los métodos rigurosos de desarrollo de programas.

• Desarrollar especificaciones de programas simples.

• Comprender los conceptos básicos de la teoría de autómatas y lenguajes.

• Comprender una teoría básica de estructuras de datos y su utilización para realizar programas simples.

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

3.1. Contenidos mínimos (según plan de estudio vigente)

Los contenidos de la materia incluyen las nociones básicas para que los alumnos puedan comprender, y usar en la práctica, el concepto de programa correcto. Para este propósito, como primer objetivo, se introduce a los alumnos a un lenguaje lógico que después se utiliza para la demostración de propiedades sobre programas. En particular, se ve un cálculo para la lógica proposicional y de primer orden., haciendo hincapié en su utilización para la resolución de problemas lógicos.

Como segundo tema se aborda el paradigma de programación funcional, los

alumnos son introducidos a los conceptos básicos de la programación funcional: modelo computacional, órdenes de reducción, evaluación lazy, el lenguaje Haskell. Además, se introduce un cálculo de programas funcionales, que permite que el alumno se inicie en el mundo de las derivaciones de programas correctos. Para esto, se hace trabajar al estudiantado con problemas de mediana complejidad, lo cual posibilita que se puedan desarrollar las habilidades matemáticas necesarias para aplicar la teoría en la práctica; además, se requiere a los estudiantes la resolución de un trabajo práctico que incluye la implementación de un programa en un lenguaje funcional avanzado, lo que permite brindar al alumno una perspectiva clara de la utilización de los lenguajes funcionales en la actualidad.

Como tercer tema se aborda la lógica de Hoare y la derivación de programas imperativos correctos. Con este fin, se introducen los conceptos de pre/postcondición, invariantes, aserciones y el transformador de predicados WP. Para lograr que los estudiantes obtengan una perspectiva adecuada de la utilización de dichas técnicas para la programación, se utilizan problemas de mediana complejidad para los cuales se obtienen programas correctos utilizando las técnicas enseñadas.

Finalmente, se brinda una introducción a los temas básicos de lenguajes y autómatas: Autómatas y lenguajes, Autómatas finitos, expresiones regulares, gramáticas y las nociones básicas de computabilidad. Estos temas proveen a los estudiantes de una visión más amplia sobre la teoría de la computación.

3.2. Ejes temáticos o unidades

Lógica y Sistemas Formales, Expresiones booleanas, cálculo proposicional, cálculo de primer orden, resolución de problemas lógicos. Inducción y recursión.
Programación Funcional, Formalismo básico. Modelo computacional. Especificación de programas funcionales. Tipos de datos: listas, árboles y pilas. Órdenes de Reducción. Evaluación Perezosa. Funciones de Orden Superior. Programación básica en el lenguaje Haskell. Verificación y Especificación de Programas Imperativos, Lógica de Hoare. Construcción de Programas Correctos. El transformador de predicados WP. Un lenguaje simple con guardas. Invariantes. Derivación de Ciclos. Metodología de Programación Dijkstra-Gries. Autómatas y Lenguajes, Autómatas Finitos. Expresiones Regulares. Gramáticas. Jerarquía de Chomsky. Nociones básicas de computabilidad.

4. ACTIVIDADES A DESARROLLAR

CLASES TEÓRICAS: Las clases teóricas planificadas (ver más abajo) se dictarán de forma presenciales. 4 horas de clases semanales. también se proveerá a los estudiantes con videos y material audiovisual para el seguimiento de la clase por medio de aulas virtuales.

CLASES PRÁCTICAS: Las clases prácticas planificadas (ver más abajo) se dictarán de forma presencial. 2 comisiones de 2 horas de clases semanales.

CLASES DE TRABAJOS PRÁCTICOS DE LABORATORIO:Las clases de laboratorio planificadas (ver más abajo) se dictarán de forma presencial. 2 comisiones de 2 horas de clases semanales.

5. PROGRAMAS Y/O PROYECTOS PEDAGÓGICOS INNOVADORES E INCLUSIVOS

6. CRONOGRAMA TENTATIVO DE CLASES E INSTANCIAS EVALUATIVAS

6.1. Cronograma de clases e instancias evaluativas.

Semana	Dia	Hora	Actividad (Tipo y Descripción)
1	Lunes	14hs	Práctico 1: Nociones básicas, recursión
1	Martes	10hs	Laboratorio: Repaso
1	Martes	16hs	Laboratorio: Repaso
1	Miércoles	14hs	Teórico: Nociones básicas
1	Jueves	8hs	Práctico 1: Nociones básicas, recursión
1	Viernes	10hs	Teórico: Nociones básicas, Programación funcional
2	Lunes	16hs	Práctica 2: Paradigma Funcional y Recursión
2	Martes	10hs	Laboratorio:Programación con Recursión: Ejercicios Básicos
2	Martes	16hs	Laboratorio: Programación con Recursión: Ejercicios Básicos
2	Miércoles	14hs	Prog.Funcional, Haskell

2	Jueves	8hs	Práctica 2: Paradigma Funcional y Recursión
2	Viernes	10hs	Teórico: Haskell
3	Lunes	14hs	Practica 3: Haskell
3	Martes	10hs	Laboratorio: Introducción Recursión en Haskell
3	Martes	16hs	Laboratorio: Introducción Recursión en Haskell
3	Miércoles	14hs	Teórico: Lógica
3	Jueves	8hs	Practica 3: Haskell
3	Viernes	10hs	Teórico: Lógica, Especificaciones
4	Lunes	14hs	Practica 3: Haskell
4	Martes	10hs	Laboratorio:Programación en Haskell: Funciones de Alto Orden y Listas
4	Martes	16hs	Laboratorio: Programación en Haskell: Funciones de Alto Orden y Listas
4	Miércoles	14hs	Teórico: Inducción
4	Jueves	8hs	Practica 3: Haskell
4	Viernes	10hs	Teórico: Derivación de Programas Funcionales
5	Lunes	14hs	Practica 4: Lógica y Especificaciones
5	Martes	10hs	Laboratorios: Programación en Haskell: Funciones de Alto Orden y Listas
5	Martes	16hs	Laboratorios: Programación en Haskell: Funciones de Alto Orden y Listas
5	Miércoles	14hs	Teórico: Derivación de Programas Funcionales
5	Jueves	8hs	Practica 4: Lógica y Especificaciones
5	Viernes	10hs	Teórico: Derivación de Programas Funcionales
6	Lunes	14hs	Parcial
6	Martes	10hs	Laboratorio: Programación en Haskell: Definiciones de Tipos Nuevos
6	Martes	16hs	Laboratorio: Programación en Haskell: Definiciones de Tipos Nuevos
6	Miércoles	14hs	Teórico: Derivación de Programas Funcionales
6	Jueves	8hs	Practica 4: Lógica y Especificaciones
6	Viernes	10hs	Teórico: Programación Imperativa, Pre y PostCondiciones

7	Lunes	14hs	Practica 5: Derivación de Programas Funcionales
7	Martes	10hs	Laboratorio: Lógica en Haskell
7	Martes	16hs	Laboratorio: Lógica en Haskell
7	Miércoles	14hs	Recuperatorio Parcial
7	Jueves	8hs	Práctica 5: Derivación de Programas Funcionales
7	Viernes	10hs	Segundo Parcial
8	Lunes	14hs	Práctica 5: Derivación de Programas Funcionales
8	Martes	10hs	Laboratorio: Derivación de Programas en Haskell
8	Martes	16hs	Laboratorio: Derivación de Programas en Haskell
8	Miércoles	14hs	Teórico: Programación Imperativa, Desarrollo de Invariantes
8	Jueves	8hs	Práctica 5: Derivación de Programas Funcionales
8	Viernes	10hs	Recuperatorio Segundo Parcial
9	Lunes	14hs	Práctica 5: Derivación de Programas Funcionales
9	Martes	10hs	Laboratorio: Derivación de Programas en Haskell
9	Martes	16hs	Laboratorio: Derivación de Programas en Haskell
9	Miércoles	14hs	Teórico: Programación Imperativa, Derivación de Programas
9	Jueves	8hs	Práctica 5: Derivación de Programas Funcionales
9	Viernes	10hs	Teórico: Programación Imperativa, Derivación de Programas
10	Lunes	14hs	Práctica 6: Programación Imperativa, Pre y PostCondiciones
10	Martes	10hs	Laboratorio: Programación Imperativa, Pre y PostCondiciones
10	Martes	16hs	Laboratorio: Programación Imperativa, Pre y PostCondiciones
10	Miércoles	14hs	Teórico: Programación Imperativa, Derivación, Ejemplos
10	Jueves	8hs	Práctica 6: Programación Imperativa, Pre y PostCondiciones
10	Viernes	10hs	Teórico: Programación Imperativa, Derivación Ejemplos
11	Lunes	14hs	Practica 7: Programación Imperativa, Invariantes

11	**Martes**	**10hs**	**Laboratorio: Programación Imperativa, Invariantes**
11	**Martes**	**16hs**	**Laboratorio: Programación Imperativa, Invariantes**
11	**Miércoles**	**14hs**	**Teórico: Programación Imperativa, Variantes y Cotas**
11	**Jueves**	**8hs**	**Practica 7: Programación Imperativa, Invariantes**
11	**Viernes**	**10hs**	**Teórico: Programación Imperativa, Fortalecimiento de Invariantes**
12	**Lunes**	**14hs**	**Practica 7: Programación Imperativa, Invariantes**
12	**Martes**	**10hs**	**Laboratorio: Programación Imperativa, Derivaciones de Programas**
12	**Martes**	**16hs**	**Laboratorio: Programación Imperativa, Derivaciones de Programas**
12	**Miércoles**	**14hs**	**Teórico: Introducción a Autómatas y Lenguajes**
12	**Jueves**	**8hs**	**Practica 7: Programación Imperativa, Invariantes**
12	**Viernes**	**10hs**	**Teórico: Introducción a Autómatas y Lenguajes**
13	**Lunes**	**14hs**	**Practica 7:Autómatas y Lenguajes**
13	**Martes**	**10hs**	**Laboratorio: Trabajo Práctico**
13	**Martes**	**16hs**	**Laboratorio: Trabajo Práctico**
13	**Miércoles**	**14hs**	**Teórico:Introducción a Autómatas y Lenguajes**
13	**Jueves**	**8hs**	**Practica 7:Autómatas y Lenguajes**
13	**Viernes**	**10hs**	**Teórico: Introducción a Computabilidad**
14	**Lunes**	**14hs**	**Practica 7:Autómatas y Lenguajes**
14	**Martes**	**10hs**	**Laboratorio: Herramientas para el procesamiento de Lenguajes**
14	**Martes**	**16hs**	**Laboratorio: Herramientas para el procesamiento de Lenguajes**
14	**Miércoles**	**14hs**	**Teórico: Introducción a Computabilidad**
14	**Jueves**	**8hs**	**Practica 7:Autómatas y Lenguajes**
14	**Viernes**	**10hs**	**Repaso de los temas vistos.**

7. BIBLIOGRAFÍA

7.1. Bibliografía obligatoria y de consulta (por lo menos algún material bibliográfico debe ser de edición 2013 o posterior).

Bibliografía Obligatoria:

• Program Construction, Calculating Implementations from Specifications. Roland Backhouse. Wiley&Sons. 2003

• Cálculo de Programas. Javier Blanco, Silvina Smith y Damián Barsotti. Facultad de Matemática , Astronomía y Física. Universidad Nacional de Córdoba. 2010

• Introduction to the Theory of Computation. Michael Sipser. PWS Pub. Co. 2012

• Aprende Haskell por el Bien de Todos, libro online, disponible en: http://aprendehaskell.es/

Bibliografía de Consulta:

• Introduction to Functional Programming using Haskell. Richard Bird. Prentice Hall Series in Computer Science. 1998

7.2. Otros: materiales audiovisuales, enlaces, otros.

Se utilizará la plataforma Google MEET para las clases virtuales. Como aula virtual se utilizará google Classroom, y para repositorios de código se utilizará Github Classroom. Las clases teóricas serán grabadas y estarán disponibles en el aula virtual, de la misma manera la resolución de problemas prototípicos serán grabadas y estarán disponibles en el aula virtual, para comunicación asincrónica se utilizará la aplicación Slack.

8. DÍA Y HORARIOS DE CLASES

Clases Teóricas:

- Miércoles 14-16hs
- Viernes 10-12hs

Clases Prácticas:

- Lunes 14-16hs
- Jueves 8-10hs

Laboratorios:

Martes 10-12hs
Martes 16-18hs

9. DÍA Y HORARIO DE CLASES DE CONSULTAS

Los días de consulta serán acordados con los alumnos.

10. REQUISITOS PARA OBTENER LA REGULARIDAD Y LA PROMOCIÓN

Evaluaciones Parciales: 2 parciales.

• Evaluación Final: El oral consta de una evaluación oral o escrita.

• CONDICIONES DE REGULARIDAD: Aprobación de los parciales y el trabajo práctico con nota de más de 5 (cinco).

11. CARACTERÍSTICAS, MODALIDAD Y CRITERIOS DE LAS INSTANCIAS EVALUATIVAS

En cada parcial se evaluarán los conocimientos obtenidos durante la materia. El examen final podrá ser escrito u oral, el cual será comprensivo y se evaluarán los conceptos dados en la materia. En caso que sea necesario los docentes podrán pedir que los alumnos resuelvan ejercicios de programación en los lenguajes de programación vistos en la materia. La materia se puede rendir de forma libre, para la cual se le podrá pedir al alumno que resuelva un trabajo práctico para demostrar el manejo de las herramientas de programación vistas en clase.

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