Universidad Nacional de Santiago del Estero

Facultad de Ciencias Exactas y Tecnologías

Carrera de Licenciatura en Sistemas de Información

PROGRAMACION LOGICA Y FUNCIONAL

PLANIFICACIÓN DE LA ASIGNATURA

AÑO 2016

Docentes

Adj.: Ing. Analía Méndez

J.T.P.: Ing. Raquel Zarco

J.T.P.: Ing. Beatriz Fernández Reuter

1. IDENTIFICACIÓN.

1.1. Nombre de la Asignatura.

Programación Lógica y Funcional.

1.2. Carrera.

Lic. en Sistemas de Información

1.3. Plan de Estudios.

Plan 2011.

1.4. Ubicación de la Asignatura.

1.4.1. Módulo - Año.

Sexto cuatrimestre - Tercer año.

1.4.2. Ciclo al que pertenece la Asignatura.

Primer Ciclo.

1.4.3. Área a la que pertenece la Asignatura.

ÁREAS	CARGA HORARIA EN HORAS RELOJ
Ciencias Básicas	--
Teoría de la Computación	--
Algoritmos y Lenguajes	75
Arquitectura, Sistemas Operativos y Redes	--
Ingeniería de Software, Bases de Datos y Sistemas de Información	--
Aspectos Profesionales y Sociales	--
Otra	--
CARGA HORARIA TOTAL DE LA ACTIVIDAD CURRICULAR	**75**

1.4.4. Carga horaria semanal.

La asignatura tiene prevista una carga horaria semanal de 5 horas, de las cuales se dedican 3 horas semanales a la formación práctica. Su cursado es de carácter cuatrimestral y su carga horaria total es de 75 horas.

1.4.5. Correlativas anteriores.

Regular: Estructura de Datos y Programación.

Aprobadas: Análisis II – Lógica II.

1.4.6. Correlativas posteriores.

Inteligencia Artificial.

1.5. Objetivos establecidos en el Plan de Estudios para la Asignatura.

El Plan de Estudios no establece objetivos específicos para la asignatura Programación Lógica y Funcional, pero cabe destacar los siguientes objetivos indicados para el área Algoritmos y Lenguajes expresados como competencias a lograr por los estudiantes y que se relacionan con la asignatura:

Desarrollar en los estudiantes competencias específicas para…

- Aplicar distintos paradigmas de programación en la resolución de problemas, empleando estructuras de control y estructuras de datos.
- La verificación de la solución de algoritmos desarrollados en los distintos paradigmas de programación.
- El uso de distintos lenguajes de programación como herramientas computacionales.

1.6. Contenidos mínimos establecidos en el Plan de Estudios para la Asignatura.

Paradigma Funcional. Clasificación. Principales características: transparencia referencial, evaluación diferida, recursividad, listas y funciones de orden superior. Estructuras de datos. Resolución de problemas y algoritmos fundamentales. Lenguajes funcionales: características fundamentales, tipos de datos y aplicaciones.

Paradigma lógico. Principales características: lógica proposicional, declaraciones, inversibilidad y backtracking. Estructuras de datos. Resolución de problemas y algoritmos fundamentales. Lenguajes lógicos: características fundamentales, tipos de datos y aplicaciones.

1.7. Año Académico.

Año 2016.

2. PRESENTACIÓN.

2.1. Ubicación de la Asignatura como tramo de conocimiento de una disciplina.

Esta asignatura corresponde al Área de Algoritmos y Lenguajes. Está orientada fundamentalmente al estudio, análisis y diseño de soluciones algorítmicas para resolver problemas computacionales a través de la programación lógica y funcional.

2.2. Conocimientos y habilidades previas que permiten encarar el aprendizaje de la Asignatura.

- Resolución de problemas con computadoras, aplicando el Paradigma Orientado a Objeto, adquiridos en Estructura de Datos y Programación.

- Conocimientos básicos de idioma extranjero adquiridos en Ingles II.
- Conocimientos adquiridos en la asignatura Lógica II.

2.3. Aspectos del Perfil Profesional del Egresado a los que contribuye la asignatura.

Posee:

- Los conocimientos básicos (lógico-matemáticos y computacionales) para una adecuada fundamentación teórica de su quehacer profesional específico.

Está capacitado para:

- Realizar tareas de investigación, tanto a nivel básico como de aplicación en el ámbito que es específico de su competencia profesional.
- Analizar y seleccionar las estructuras de datos, necesarias para los diferentes Sistemas de Información.

3. OBJETIVOS.

Que el alumno desarrolle las siguientes competencias básicas, transversales y específicas.

3.1 Competencias básicas

- Capacidad de investigación.
- Capacidad para actuar en nuevas situaciones.
- Capacidad para buscar, seleccionar y procesar la información necesaria para la resolución de problemas.
- Capacidad para verificar las soluciones encontradas.

3.2 Competencias transversales

- Capacidad para relacionar la teoría a la práctica.
- Capacidad para apropiar nuevos métodos, paradigmas y tecnologías y adaptarse a nuevas situaciones.
- Capacidad para identificar, plantear y resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad.
- Capacidad para la comunicación oral y escrita enfocada en el intercambio y transmisión de los conocimientos.

3.3 Competencias Especificas

- Conocer las principales fortalezas y debilidades del paradigma de programación declarativa.
- Capacidad para aplicar los principios y técnicas de la programación declarativa.

- Capacidad para reconocer y diferenciar el estilo de programación declarativa respecto del estilo de programación imperativa.
- Capacidad para realizar la búsqueda creativa de soluciones algorítmicas y la selección criteriosa de la alternativa más adecuada aplicando los mecanismos propios de los lenguajes de programación declarativa.
- Destreza para diseñar, codificar, ejecutar y depurar, e interpretar programas usando un lenguaje de programación funcional, como Haskell.
- Destreza para diseñar, codificar, ejecutar y depurar, e interpretar programas usando un lenguaje de programación lógica, como Prolog.
- Capacidad para trabajar con definiciones de funciones mediante programas, tanto funcionales como lógicos.
- Destreza en el desarrollo de especificaciones de programas simples aplicando programación declarativa.
- Capacidad para verificar la solución de algoritmos desarrollados usando programación declarativa.

4. SELECCIÓN Y ORGANIZACIÓN DE CONTENIDOS.

4.1. Programa Sintético sobre la base de los contenidos mínimos.

Unidad I: Introducción al Paradigma de Programación Declarativa

Paradigmas. Clasificación.

Unidad II: Paradigma Funcional

Paradigma Funcional. Clasificación. Principales características. Transparencia referencial. Recursividad. Listas. Estructuras de datos.

Unidad III: Programación Funcional

Lenguajes funcionales. Características fundamentales. Tipos de datos. Funciones de orden superior. Evaluación diferida o perezosa. Aplicaciones de los lenguajes funcionales. Resolución de problemas y algoritmos fundamentales.

Unidad IV: Paradigma Lógico

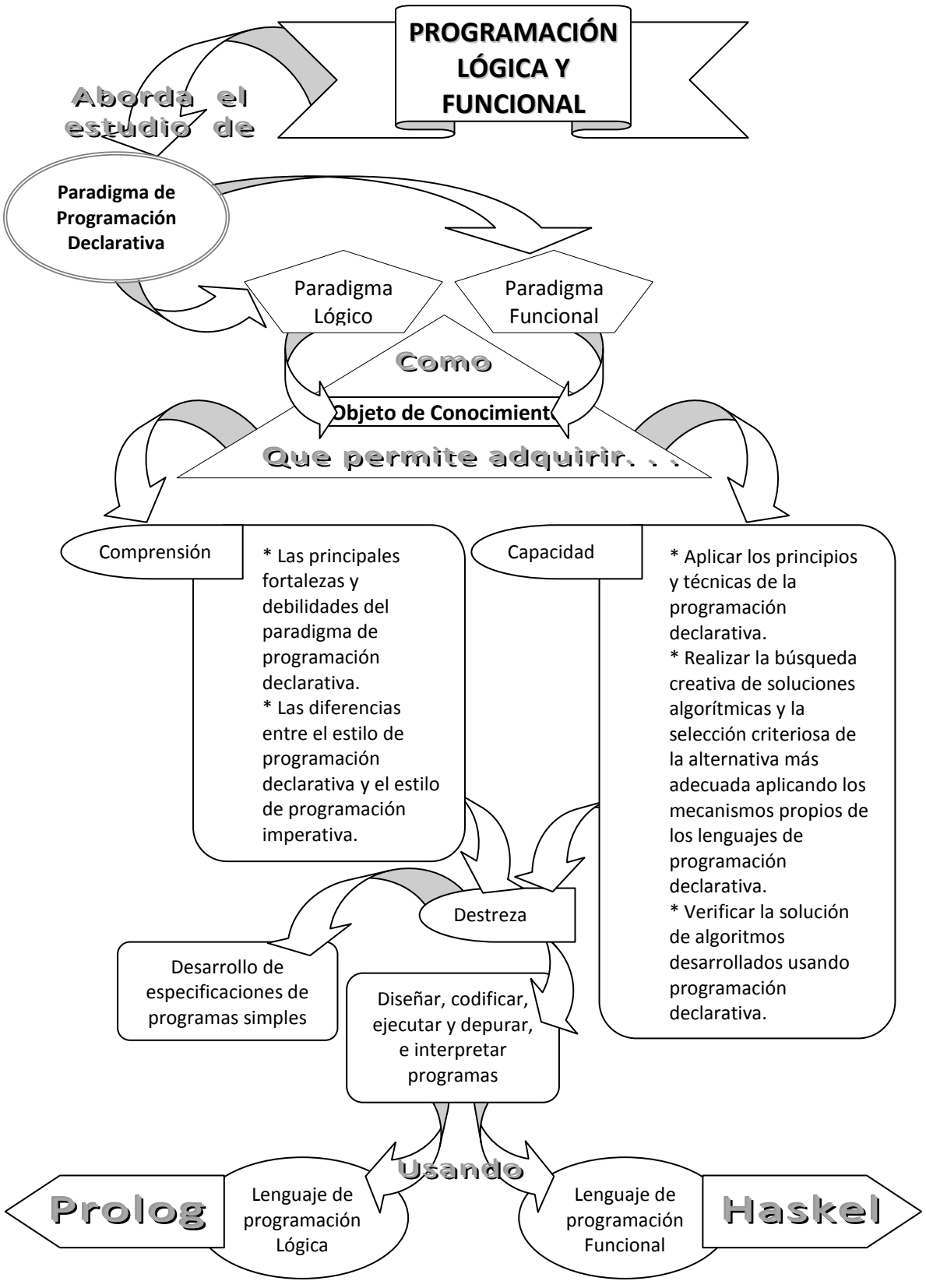
Paradigma lógico. Principales características: lógica proposicional, declaraciones, inversibilidad, backtracking. Estructuras de datos.

Unidad V: Programación Lógica

Lenguajes lógicos. Características fundamentales, tipos de datos. Aplicaciones de los lenguajes lógicos. Resolución de problemas y algoritmos fundamentales.

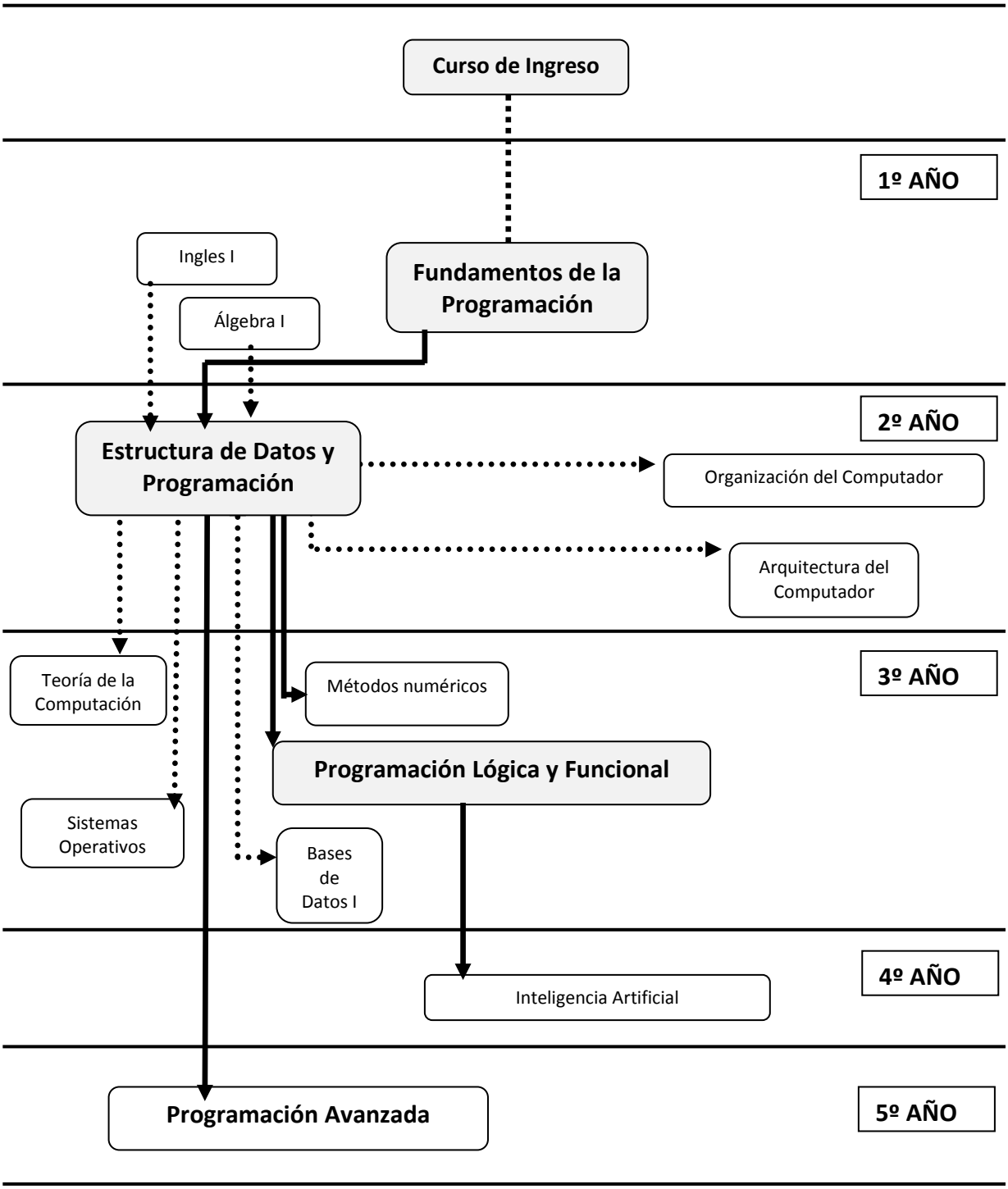
4.2. Articulación Temática de la Asignatura /Obligación Curricular

En la figura 1 se presentan los principales conceptos a tratar en la asignatura y la relación entre los mismos.



4.3. Integración horizontal y vertical con otras asignaturas.

Esta asignatura corresponde al Área Algoritmos y Lenguajes, integrado además por las asignaturas Fundamentos de la Programación, Estructura de Datos y Programación; Programación Avanzada, y Métodos Numéricos. En la figura se muestra la integración mencionada y la vinculación de las asignaturas del área.



Esta Área tiene como objetivos desarrollar en los estudiantes competencias específicas para:

- La búsqueda creativa de solución/es algorítmica/s a un problema, y la selección criteriosa de la alternativa más adecuada, aplicando distintos paradigmas de programación.
- El empleo de estructuras de control y estructuras de datos en la resolución de problemas, aplicando distintos paradigmas de programación
- La verificación de la solución de algoritmos desarrollados en los distintos paradigmas de programación.
- El uso distintos lenguajes de programación como herramienta computacional.

En Fundamentos de la Programación, el alumno adquiere los conocimientos básicos de la programación, resuelve problemas y usa un lenguaje de programación como herramienta, aplicando el Paradigma de Programación Imperativo.

En Estructura de Datos y Programación, el alumno adquiere conocimientos de estructuras de datos avanzadas y del paradigma de programación orientada a objetos.

En Programación Lógica y Funcional y en Programación Avanzada se avanza en el aprendizaje de otros paradigmas de programación de manera de lograr que el alumno formalice y extienda sus conocimientos con respecto a la programación y en Métodos Numéricos adquiere los conocimientos sobre algoritmos numéricos y propagación de error.

4.4. Programa Analítico

Unidad I: Introducción al Paradigma de Programación Declarativa

Paradigmas de programación. Concepto. Clasificación. Paradigmas declarativos: paradigma funcional y paradigma lógico.

Unidad II: Paradigma Funcional

Paradigma funcional: definición. Clasificación. Principales características. Concepto de programa en el paradigma funcional. Concepto de función. Definición de funciones. Estructuras de datos. Tuplas. Listas. Expresiones Lambda. Polimorfismo y tipos genéricos. Recursividad. Transparencia referencial.

Unidad III: Programación Funcional

Lenguajes funcionales. Características fundamentales. Ventajas de los lenguajes funcionales. Lenguaje de programación Haskell. Tipos de datos. Notación bidimensional. Módulos. Notación de listas. Notación de listas por comprensión. Operadores infijos y prefijos. Reglas de precedencia. Funciones para el manejo de listas. Funciones para el

manejo de tuplas. Funciones de orden superior. Formas de evaluación. Evaluación diferida o perezosa. Aplicaciones de los lenguajes funcionales. Resolución de problemas. Algoritmos fundamentales.

Unidades IV: Paradigma Lógico

Paradigma lógico: definición. Principales características. Lógica proposicional. Fundamentación. Predicados. Razonamiento y silogismos. Relaciones, hechos y reglas. Definición de programa en paradigma lógico. Diferencia entre una función y una relación. Declaraciones. Concepto de variable o incógnita. Unificación. Backtracking. Inversibilidad. Estructuras de datos: listas. Pattern matching.

Unidades V: Programación Lógica

Lenguajes lógicos. Características fundamentales. Ventajas de los lenguajes lógicos. Lenguaje de programación Prolog. Estructura de un programa en Prolog. Tipos de datos. Predicadores de orden superior. Functores. Polimorfismo. Aplicaciones de los lenguajes lógicos. Resolución de problemas. Algoritmos fundamentales.

4.5. Programa y Cronograma de Trabajos Prácticos

No se prevé.

4.6. Programa y Cronograma de Actividades de Formación Experimental

4.6.1. Taller de Lenguaje de Programación Haskell.

Contenidos

Introducción al entorno de programación del lenguaje Haskell. Expresiones aritméticas y prioridad de operadores. Listas. Listas por comprensión. Tuplas. Tipos de datos. Sintaxis: Patrones, Guardas, Cláusula Where, Bindings, Expresiones Case. Recursión. Funciones de orden superior: Currificadas, Maps y Filters, Lambdas, Folds, Composición de funciones. Módulos. Tipos definidos por el usuario. Entrada y Salida.

Objetivos.

Que los alumnos logren…

- Aplicar los principios y técnicas de la programación funcional.
- Diseñar, codificar, ejecutar, depurar, e interpretar programas usando el lenguaje de programación funcional Haskell.
- Verificar la solución de algoritmos desarrollados usando programación funcional.
- Realizar la búsqueda creativa de soluciones algorítmicas y la selección criteriosa de la alternativa más adecuada aplicando los mecanismos propios del lenguaje de programación funcional Haskell.

Metodología

Este taller será de carácter teórico práctico, con actividades prácticas guiadas, resolución de ejercicios de programación en aula, y el desarrollo de proyectos domiciliarios. Las clases se desarrollarán en el Laboratorio de Informática.

Evaluación

Para aprobar el Taller de lenguaje de programación Haskell el alumno deberá presentar y defender satisfactoriamente un proyecto de programación. La escala de valoración a emplear será cualitativa dicotómica (aprobado – desaprobado).

Criterios de evaluación y aprobación

- Buen nivel de aplicación de los principios y técnicas de la programación funcional estudiados.
- Correcta resolución del problema abordado y adecuada exposición de las conclusiones obtenidas.
- Capacidad para el análisis crítico.

Instrumentos para la actividad

Para desarrollar esta actividad de formación experimental es necesario contar con los siguientes recursos:

- Espacio de la asignatura en la plataforma CUV (Centro Universitario Virtual).
- Grupo de comunicación e intercambio en la red virtual Facebook.
- Laboratorio de computadoras con la instalación de:
  - Lenguaje de programación Haskell.
  - Interfaz WinGHCI de Haskell para Windows.

Cronograma

		Meses	
Actividad	**Tema**	**Agosto**	**Septiembre**
1	Introducción al entorno de programación del lenguaje Haskell. Expresiones aritméticas y prioridad de operadores	2º Semana	---
2	Listas. Listas por comprensión. Tuplas. Tipos de datos	3º Semana	---
3	Sintaxis: Patrones, Guardas, Cláusula Where, Bindings, Expresiones Case	4º Semana	---
4	Recursión	5º Semana	---
5	Funciones de orden superior: Currificadas, Maps y Filters, Lambdas, Folds, Composición de funciones	---	2º Semana
6	Módulos. Tipos definidos por el usuario	---	3º Semana

		Meses	
Actividad	**Tema**	**Agosto**	**Septiembre**
7	Entrada y Salida	---	4º Semana
8	Evaluación	---	5º Semana

Esta actividad experimental se desarrollará en un módulo con una dedicación de sesiones semanales de tres (3) horas, totalizando en el año académico 24 horas.

4.6.2. Taller de Lenguaje de Programación Prolog.

Contenidos

Introducción a Prolog. Elementos del lenguaje: variables, términos, operadores, hechos, reglas. Cláusulas de Horn. Tipos de Datos simples. Objetos de datos estructurados. Concepto y utilización de las listas. Recursión. El corte en Prolog. Predicados de control. Unificación. Entrada/salida.

Objetivos.

Que los alumnos logren…

- Aplicar los principios y técnicas de la programación lógica.
- Diseñar, codificar, ejecutar, depurar, e interpretar programas usando el lenguaje de programación lógica Prolog.
- Verificar la solución de algoritmos desarrollados usando programación lógica.
- Realizar la búsqueda creativa de soluciones algorítmicas y la selección criteriosa de la alternativa más adecuada aplicando los mecanismos propios del lenguaje de programación lógica Prolog.

Metodología

El desarrollo del taller será de carácter teórico práctico, con actividades prácticas guiadas, resolución de ejercicios de programación en aula, y el desarrollo de proyectos domiciliarios. Las clases se desarrollarán en el Laboratorio de Informática.

Evaluación

Para aprobar el Taller de lenguaje de programación Prolog el alumno deberá presentar y defender satisfactoriamente un proyecto de programación. La escala de valoración a emplear será cualitativa dicotómica (aprobado – desaprobado).

Criterios de evaluación y aprobación

- Buen nivel de aplicación de los principios y técnicas de la programación lógica estudiados.

- Correcta resolución del problema abordado y adecuada exposición de las conclusiones obtenidas.
- Capacidad para el análisis crítico.

Instrumentos para la actividad

Para desarrollar esta actividad de formación experimental es necesario contar con los siguientes recursos:

- Espacio de la asignatura en la plataforma CUV (Centro Universitario Virtual).
- Grupo de comunicación e intercambio en la red virtual Facebook.
- Laboratorio de computadoras con la instalación de:
  - Lenguaje de programación SWI-Prolog.

Cronograma

		Meses	
Actividad	**Tema**	**Octubre**	**Noviembre**
1	Introducción a Prolog. Elementos del lenguaje: variables, términos, operadores, hechos, reglas. Cláusulas de Horn	1° Semana	---
2	Tipos de Datos simples. Objetos de datos estructurados	2° Semana	---
3	Concepto y utilización de las listas. Recursión	3° Semana	---
4	El corte en Prolog. Predicados de control	4° Semana	---
5	Unificación	---	1° Semana
6	Entrada/salida	---	2° Semana
7	Evaluación	---	3° Semana

Esta actividad experimental se desarrollará en un módulo con una dedicación de sesiones semanales de tres (3) horas, totalizando en el año académico 21 horas.

5. BIBLIOGRAFIA.

5.1. Bibliografía Específica.

Título	Autor/es	Editorial	Año y Lugar de edición	Disponible en	Cant. Ejemplares disponibles
Razonando con Haskell	Ruiz Blas, Gallardo Jose, Guerrero Pablo, Gutierrez Francisco	Editorial PARANINFO	2007, España	Biblioteca del Departamento de Informática	1
Programación Lógica: Teoría y practica	Pascual Julián - Iranzo, Maria Alpuente	Prentice Hall	2007, España		1

Título	Autor/es	Editorial	Año y Lugar de edición	Disponible en	Cant. Ejemplares disponibles
Programming in Prolog Using the ISO Standard	William F. Clocksin, Christopher S. Mellish	Springer	2003 EEUU		1
Introducción a la Programación funcional con Haskell	Richard Bird	Prentice Hall	2000 España		1
Lógica para la computación	Luis de Ledesma	AlfaOmega	2010, Mexico		1

5.2. Bibliografía General o de Consulta.

5.2.1. Referencias web.

Titulo	Dirección
Página principal del Lenguaje Haskell	http://www.haskell.org
Una introducción agradable a Haskell	http://www.lcc.uma.es/~blas/pfHaskell/gentle/
Haskell Reference	http://zvon.org/other/haskell/Outputglobal/index.html
Página principal de swi-prolog	http://www.swi-prolog.org
A Guide to Functional Programming on the Web	http://homepages.inf.ed.ac.uk/wadler/guide.html
The Haskell Programming Language	https://wiki.haskell.org/Haskell
¡Aprende Haskell por el bien de todos!	http://aprendehaskell.es/
Guia rápida de Haskell	https://cloud.github.com/downloads/jsoffer/cheatsheet/CheatSheetEs.pdf

5.2.2. Bibliografía digital.

Titulo	Autor/es	Editorial	Edición
Logic, Programming and Prolog (2ed)	Ulf Nilsson y Jan Maluszynski	John Wiley & Sons Ltd.	2000
Disponible en: http://www.ida.liu.se/~ulfni/lpp/			
El lenguaje de programacion Prolog	Francisco Toledo Lobo - Julio Pacheco Aparicio - M. Teresa Escrig Monferrer	Grupo de investigación Intelligent Control Systems. Universidad	2001

Titulo	Autor/es	Editorial	Edición
		Jaume I	
Disponible en: http://mural.uv.es/mijuanlo/PracticasPROLOG.pdf			
Programación Funcional	Jeroen Fokker	Departamento de Informática, Universidad de Utrecht	1996
Disponible en: http://www.staff.science.uu.nl/ fokke101/courses/fp-sp.pdf			
RAZONANDO CON HASKELL. Un curso sobre programación funcional	Blas C. Ruiz - Francisco Gutiérrez - Pablo Guerrero - José E. Gallardo. E.T.S.I. Informatica. Universidad de M álaga	International Thomson Editores Spain Paraninfo	2004
Disponible en: http://www.lcc.uma.es/~pepeg/pfHaskell/index.html			
SWI-Prolog 5.0. Reference Manual	Jan Wielemaker	University of Amsterdam. Dept. of Social Science Informatics (SWI)	2002
Disponible en: http://www.uco.es/users/ma1fegan/Comunes/manuales/lia/swirefman.pdf			
Programacion Funcional	Mario Rodriguez Artalejo	Universidad Complutense de Madrid	2014
Disponible en: http://gpd.sip.ucm.es/sonia/docencia/pf/cursoPF.pdf			
Learn Prolog Now!	Patrick Blackburn, Johan Bos - Kristina Striegnitz		2001
Disponible en: http://www.cmpe.boun.edu.tr/sites/default/files/learn_prolog_now.pdf			
http://www.learnprolognow.org/lpnpage.php?pageid=online			
Programación Declarativa	José Carpio Cañada - Gonzalo Antonio Aranda Corral - José Marco de la Rosa	Universidad de Huelva. Servicio de Publicaciones	2010
Disponible en: http://www.uhu.es/jose.carpio/N_95.pdf			

6. ESTRATEGIAS METODOLOGICAS.

6.1. Aspectos pedagógicos y didácticos.

La metodología de enseñanza que se aplicará en el aula será llevada a cabo en un espacio de diálogo y construcción, en el que se trabaje interactuando permanentemente, y de esta manera tanto los alumnos como el docente se consideran fuente de información.

En las clases teóricas se aplicará el método expositivo / lección magistral donde la exposición explicativa se realizará con el diseño de las clases en PowerPoint o con el uso del pizarrón, usando como recursos didácticos: esquemas, tablas, gráficos, etc. Cada contenido desarrollado es mediado para su mejor comprensión y con el fin de propiciar el

diálogo y discusión; además se realizan ejercicios de aplicación con el fin de aclarar conceptos, técnicas y métodos a utilizar.

Las exposiciones teóricas se complementan con el desarrollo del Taller de Lenguaje de Programación Haskell y el desarrollo del Taller de lenguaje de programación Prolog a los efectos de lograr una práctica de formación experimental que permita encontrar las soluciones a los problemas planteados en las clases teóricas.

En la práctica se utiliza software libre; se procura lograr el desarrollo de habilidades y capacidades prácticas, y preparar a los alumnos para enfrentar los distintos cambios del mundo tecnológico.

Las actividades de estudio tanto prácticas como de tipo teórico serán apoyadas y desarrolladas mediante la plataforma Moodle del Centro Universitario Virtual (CUV). Mediante esta plataforma se utilizarán herramientas tales como wikis y cuestionarios evaluativos y de diagnóstico, así como la presentación digital de las actividades y proyectos propuestos en las prácticas de formación experimental.

En el desarrollo de las actividades áulicas se destaca el uso de las instalaciones del laboratorio de computadoras, tanto para llevar a cabo ejercicios prácticos respectivos a los temas en estudio así como para el uso de herramientas de apoyo didáctico.

Las actividades áulicas se caracterizarán principalmente por una integración entre los conceptos teóricos impartidos y su aplicación en ejercicios prácticos.

6.2. Actividades de los alumnos y de los docentes.

6.2.1. Actividades de los docentes:

Actividades del Profesor Adjunto: Desarrollar clases teórico - prácticas. Atender consultas de los alumnos. Coordinar el equipo cátedra. Seleccionar bibliografía y preparar material didáctico. Evaluar permanentemente. Supervisar el desarrollo de las clases prácticas y la preparación de las actividades de formación experimental. Supervisar y coordinar actividades en plataforma del CUV.

Actividades de Auxiliares Docentes: Coordinar las actividades de formación experimental. Desarrollar las clases prácticas en el Laboratorio de Informática. Coordinar las actividades de los alumnos en plataforma del CUV. Elaborar materiales correspondientes a los talleres. Colaborar en la preparación de material didáctico. Atender consultas de alumnos. Colaborar y participar en el proceso de evaluación.

6.2.2. Actividades de los alumnos

Las actividades de los alumnos son:

- Asistir y participar de las clases de la asignatura.

- Efectuar seguimiento de las actualizaciones en el espacio de la asignatura en la plataforma CUV y completar las actividades tales como wikis, cuestionarios, etc.
- Desarrollar las prácticas propuestas por la cátedra y presentarlas en tiempo y forma.
- Realizar revisión y consulta bibliográfica para profundizar los temas estudiados en clase.
- Conformar grupos de estudio con actitud colaborativa y con una participación activa y dedicada.
- Integrar, participar y colaborar en las actividades grupales que se propongan.
- Participar y completar las actividades de formación experimental propuestas.
- Presentar en tiempo y forma el trabajo solicitado en la evaluación integradora.

6.3. Mecanismos para la integración de docentes.

Se plantea principalmente la integración con la asignatura Estructuras de Datos y Programación. Durante el desarrollo de dicha asignatura los alumnos cursan un Taller de programación en el lenguaje Java, aprendiendo el paradigma de Programación Orientada a Objetos.

La asignatura Estructuras de Datos integra el segundo año de la carrera de Licenciatura en Sistemas de Información y capacita al alumno en el tratamiento de estructuras de datos así como respecto a los principales conceptos referidos al desarrollo de programación en computadoras, tales como: Estructuras de datos encadenadas, listas; Recursividad; Estrategias de diseño de algoritmos; el Paradigma de programación orientada a objetos, abarcando entre otros conceptos polimorfismo y herencia; y la Resolución de problemas y algoritmos.

La coordinación entre las asignaturas se realiza mediante encuentros anuales en los que se revisan los logros alcanzados en la comprensión y aplicación del paradigma de programación estudiado y las actividades prácticas.

6.4. Cuadro sintético.

Teóricas	Formación Practica					
	Formación Experimental	Resolución de problemas del mundo real	Actividades de proyectos y diseño de Sistemas de Información	Instancias supervisadas de formación en la práctica profesional	Otras	Total
30	45	--	--	--	--	45

6.5. Recursos didácticos.

Se prevé la utilización de recursos didácticos: pizarra, libros, apuntes, Internet, plataforma Moodle del Centro Universitario Virtual (CUV), laboratorio de informática, diapositivas, retroproyector o videoproyector y softwares, tales como:

- Lenguaje Haskell.
- Interfaz WinGHCI de Haskell para Windows.
- Lenguaje Prolog.
- Lenguaje de programación SWI-Prolog.
- Sistema operativo Windows.

7. EVALUACIÓN.

7.1. Evaluación diagnóstica.

Teniendo en cuenta que la evaluación diagnóstica no sólo es una estimación, sino que tiene como propósito contribuir al aprendizaje, se llevará a cabo una única evaluación diagnóstica, al comienzo de las clases, cuya finalidad será determinar el nivel de conocimientos y habilidades previas que permitan encarar el aprendizaje de la asignatura.

Se evalúan contenidos adquiridos en las asignaturas Lógica II y Estructura de Datos y Programación.

La evaluación diagnóstica será especialmente diseñada, individual, escrita y objetiva. Se hará una prueba de opción múltiple para que el alumno marque la opción correcta. El nivel de calificación será cualitativa politómica (nivel medio, bajo o alto).

7.2. Evaluación formativa.

La evaluación formativa es de carácter continuo y está dirigida a evaluar el proceso de enseñanza-aprendizaje durante todo el desarrollo de la asignatura. Para ello se tendrá en cuenta el desempeño que demuestren los alumnos en la realización de las actividades planteadas, a fin de encarar, si fuera necesario, acciones correctivas.

Se aplicarán herramientas disponibles a través de la plataforma del CUV (Centro de Universitario Virtual). En particular se utilizarán herramientas wiki y cuestionario.

Esta modalidad de evaluación permitirá identificar la evolución en el aprendizaje de los alumnos y el grado de impacto de la propuesta educativa que lleva a cabo la cátedra. Se evaluará:

- Participación del alumno en clase y en las actividades.
- Disposición y desempeño del alumno en la resolución de las actividades propuestas.

- Presentación en tiempo y forma de las actividades propuestas.
- Capacidad de resolución y de análisis de los problemas de carácter teórico y práctico que se le presenten al alumno.

7.2.1. Programa y Cronograma de Evaluaciones Formativas.

		MESES			
ACTIVIDAD	**TEMA**	**Agosto**	**Septiembre**	**Octubre**	**Noviembre**
Wiki 1	Principales características del Paradigma Funcional	3° y 4° Semanas	---	---	
Cuestionario Domiciliario 1	Características fundamentales de Haskell como Lenguaje Funcional	5° Semana	1°, 2° y 3° Semanas	---	
Wiki 2	Principales características del Paradigma Lógico	---	---	2° y 3° Semanas	
Cuestionario Domiciliario 2	Características fundamentales de Prolog como Lenguaje de Lógico	---	---	4° Semana	1° y 2° Semanas

7.3. Evaluación parcial.

7.3.1. Programa y Cronograma de Evaluaciones Parciales.

		Meses				Fecha
Parciales	**Tema**	**Agosto**	**Septiembre**	**Octubre**	**Noviembre**	
Evaluación Parcial 1	Paradigma Funcional	5° Semana	---	---	---	29/08
Recuperatorio Eval.1		---	3° Semana	---	---	12/09
Evaluación Parcial 2	Paradigma Lógico	---	---	4° Semana	---	24/10
Recuperatorio Eval. 2		---	---	---	2° Semana	07/11

7.3.2. Criterios de evaluación.

	Modalidad	Objetivos	Criterios de Evaluación
Evaluación Parcial 1	Cuestionario Multiple Choice Presencial.	Determinar si los alumnos han adquirido los conocimientos suficientes relativos a: * Las principales características y las fortalezas y debilidades del paradigma de programación funcional. * Las diferencias entre el estilo de programación funcional y el estilo de programación imperativa.	Capacidad para reconocer aspectos de relevancia en la temática planteada. Habilidad para relacionar, contrastar y vincular los contenidos estudiados.
Recuperatorio Eval.1			
Evaluación Parcial 2		Determinar si los alumnos han adquirido los conocimientos suficientes relativos a: * Las principales características y las fortalezas y debilidades del paradigma de programación lógica. * Las diferencias entre el estilo de programación lógica y el estilo de programación imperativa.	
Recuperatorio Eval.2			

7.3.3. Escala de Valoración, instancias de recuperación y condiciones de promoción.

La escala de valoración a emplear para las evaluaciones parciales y los recuperatorios será cuantitativa politómica. Escala: 1 al 100.

El puntaje mínimo para aprobar los parciales es de cincuenta (50) puntos, sobre una calificación máxima de 100.

Se otorga una oportunidad de recuperación para cada parcial en caso de desaprobación.

Actividad	Escala de Valoración	Recuperación
Wikis	Cualitativa dicotómica (aprobado – desaprobado)	Recuperatorio del tema correspondiente
Actividades de Taller		Completar o corregir las actividades prácticas
Talleres		
Cuestionarios Evaluativos Domiciliarios	Cuantitativa politómica. Escala: 1 al 100. El puntaje mínimo para aprobar es de cincuenta (50) puntos, sobre una calificación máxima de 100.	Recuperatorio del tema correspondiente
Parciales Teóricos		

7.4. Evaluación integradora.

La evaluación integradora se llevará a cabo a través de un Trabajo Integrador Final en el que los alumnos deberán generar un informe respecto a las conclusiones a las que arribaron mediante las prácticas realizadas aplicando programación funcional y programación lógica, incluyendo una comparación entre ambas.

7.5. Autoevaluación

La autoevaluación de la asignatura desde la perspectiva de los alumnos, se llevará a cabo a través de encuesta de respuestas cerradas implementada al finalizar el cursado.

La autoevaluación de la asignatura desde la perspectiva de los docentes se llevará a cabo a partir de los resultados obtenidos en las evaluaciones y condiciones finales de cursado.

7.6. Evaluación sumativa.

7.6.1. Condiciones para lograr la promoción sin Examen Final de la Asignatura.

No corresponde.

7.6.2. Condiciones para lograr la regularidad de la Asignatura.

a) Aprobar Wikis, o el recuperatorio del tema correspondiente.

b) Todo alumno que no apruebe Wikis, deberá aprobar el recuperatorio del tema correspondiente.

c) Presentar todas las actividades de los talleres y aprobar los proyectos propuestos en las prácticas de formación experimental.

d) Aprobar los Cuestionarios Evaluativos Domiciliarios y las evaluaciones parciales de teoría o sus correspondientes recuperatorios.

e) Presentación en tiempo y forma de la evaluación integradora.

7.7. Examen final.

La evaluación final será escrita u oral sobre los temas incluidos en la programación de la asignatura.

7.8. Examen libre.

Primera Etapa: Presentación, prueba y defensa de un planteamiento práctico. El trabajo deberá ser presentado en soporte digital e impreso, cumplimentando las pautas establecidas en el enunciado, la presentación deberá realizarse al término del plazo de días hábiles indicado a partir de la entrega del enunciado.

Segunda Etapa: Aprobar una evaluación oral/escrita de contenidos teóricos del programa analítico.

Tercera Etapa: Aprobar una evaluación en computadora aplicando los contenidos teóricos y prácticos de la asignatura.

------0ooo><ooo0------

Ing.Méndez
Docente a cargo