Ministerio de Cultura y Educación **(Programa del año 2017)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area IV: Pr. y Met. de Des. del Soft.

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
PROGRAMACION II	ING. EN COMPUT.	28/12 026/1	2017	2° cuatrimestre
PROGRAMACION II	ING. INFORM.	2-08/15	2017	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
BERON, MARIO MARCELO	Prof. Responsable	P.Adj Exc	40 Hs
MIRANDA, ENRIQUE ALFREDO	Responsable de Práctico	JTP Simp	10 Hs
CASTILLO ELÍAS, SANTIAGO	Auxiliar de Práctico	A.2da Simp	10 Hs
FERRARI, AGUSTIN	Auxiliar de Práctico	A.2da Simp	10 Hs
PEREZ, NORMA BEATRIZ	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	1 Hs	3 Hs	6 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	2º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
07/08/2017	18/11/2017	15	90

IV - Fundamentación

Los paradigmas de programación son uno de los núcleos basales de los profesionales en informática. Cada paradigma de programación proporciona métodos, estilos de programación, etc. que facilitan el abordaje de problemas informáticos. Por esta razón, impartir los conceptos relacionados con esta temática es primordial en las carreras de ingeniería. Uno de los paradigmas que ha adquirido mucho énfasis en los últimos tiempos, debido a sus fundamentos y a los métodos y herramientas que proporciona para la solución de problemas, es la Programación Orientado a Objetos (POO).
POO proporciona mecanismos interesantes que libran al programador de detalles de implementación que incrementan la posibilidad de introducir errores de programación. Además, posibilita que el código desarrollado pueda ser reutilizado evitando de esta manera la pérdida de tiempo producida por la re-implementación de soluciones. Otro paradigma reciente que ha alcanzado mucha importancia es el proporcionado por la Programación Dirigida por Eventos. Esta forma de concebir la programación tiene como principal objetivo proporcionar los métodos y herramientas adecuadas para desarrollar sistemas donde el flujo de ejecución está determinado por el usuario, acciones internas del sistema u otros externos. El creador de un programa dirigido por eventos debe definir los eventos que manejarán su programa y las acciones que se realizarán al

producirse cada uno de ellos, lo que se conoce como el administrador de evento. Los eventos soportados estarán determinados por el lenguaje de programación utilizado, por el sistema operativo e incluso por eventos creados por el mismo programador. La Programación Dirigida por Eventos es la base para desarrollar interfaces de usuario aunque también es útil para desarrollar otros tipos de aplicaciones como por ejemplo: módulos del núcleo de un sistema operativo. La programación basada en scripts es importante en el ámbito de procesamiento por lotes y archivos de comando en sistemas operativos y así como también su evolución hacia lenguajes de propósito general. Como se puede observar en las breves descripciones presentadas en los párrafos precedentes, los paradigmas Orientados a Objetos, Dirigidos por Eventos y la programación Script son fundamentales para el desarrollo de aplicaciones industriales y científicas y por consiguiente son tópicos muy importantes que deben conformar el trasfondo de conocimientos de todo Ingeniero.

V - Objetivos / Resultados de Aprendizaje

Al finalizar el curso se espera que el alumno sea capaz de:
- Conocer y aplicar correctamente los conceptos fundamentales del paradigma de programación orientada a objetos (POO).
- Desarrollar una visión clara del tipo de situaciones en las que el paradigma POO es adecuado y la forma en que los conceptos de clase, herencia, polimorfismo y ligadura dinámica de mensajes interactúan.
- Desarrollar y ejecutar programas cortos usando un lenguaje representativo del POO.
- Comprender los conceptos fundamentales de la programación dirigida por eventos (PDE).
- Aplicar esta forma de programación en ejemplos concretos que involucren eventos y manejadores de eventos.
- Comprender los conceptos fundamentales de la programación de la programación basada en scripts .

VI - Contenidos

Unidad I

Evolución desde “Tipo de Datos Abstractos” a “Programación Orientada a Objetos”. Sintaxis y semántica de Java. Definición de variables. Visibilidad y Vida de las variables. Tipos de dato primitivos. Conversión de tipos. Uso de Operadores mediante ejemplos sencillos en laboratorio: Aritméticos, Asignación, instanceof, condicional, incrementales, relacionales, lógicos, concatenación de caracteres. Precedencia entre operadores. La clase String. Integer, Float y Number. Arreglos. Wrappers. Estructuras de control de flujo: Selección if else, Selección switch, Bucle while, Bucle for, Bucle do while, Sentencia break, Sentencia continue. Comentarios en Java. Caracteres especiales.

Unidad II

Programación Orientada a Objetos y Java. ¿Qué es una clase?
Ejemplos de construcción de una clase en Java. Clases y Objetos en Java. Atributos. Métodos en Java. Pasaje de parámetros. Encapsulamiento. Control de acceso. Constructores. Herencia en Java. Sub-clases y super-clases. This. Super. Herencia múltiple. Redefinición de métodos heredados. Accesibilidad a clases o interfaces, variables, Polimorfismo. Ligadura Dinámica. Clase abstracta. Abstracción en Java. Interfaz; ejemplos en Java. Métodos Abstractos. Paquetes. Relaciones entre paquetes. Clases Genéricas.

Unidad III

El framework Collection en Java: intefaces, implementaciones y algoritmos. La clase Vector. La clase Stack.
La interfaz Collection. Conjuntos. La interfaz Set y sus implementaciones. La interfaz List y sus implementaciones. Iteradores. La interfaz Map y sus implementaciones. La clase Collections. La Interfaz Cloneable y Serializable.

Unidad IV

Streams. Tipos de Streams. Orientados a caracteres. Orientados a bytes. File Streams. FileReader. FileWriter. FileInputStream & FileOutputStream. Buffers. BufferedReader. BufferedWriter. BufferedInputStream. BufferedOutputStream.

Unidad V

Manejo de Excepciones. Ventajas. Manejo de excepciones en Java: La clase Throwable; bloques try, catch y finally. Tipos de Excepciones: Excepciones verificadas y no verificadas. Creación de excepciones propias. La sentencia throw. La clase Exception como superclase. La cláusula throws. Estudio comparativo y mediante ejemplos en laboratorio.

Unidad VI

La interfaz gráfica. AWT. Concepto y uso. Componentes y eventos soportados. Jerarquía de componentes, eventos y sus

relaciones Estructura de una aplicación AWT estudiada mediante ejercicios en laboratorio. Swing. Estructura de una aplicación Swing. Swing vs. AWT. Componentes Swing: Contenedores. JFrame. JComponent JDialog. JApplet. JPanel. Organización en el IDE. Manejo de ventanas y layout. Introducción a la Programación dirigida a Eventos. El modelo de eventos de Java. Programación de eventos GUI en Java.

Unidad VII
El Paradigma de Programación dirigida a Eventos. Diferencias entre las distintas formas de paradigmas de programación vistos. El usuario dirigiendo el flujo del programa. Detección de eventos. Problemática. Interfaces Gráficas con el Usuario en un entorno dirigido a Eventos. Herramientas visuales de desarrollo. Lenguajes Utilizados en la Programación dirigida a Eventos. Referencias. Enlaces externos. Definición de Eventos y notificaciones. Subscripción de Eventos. Proceso de Suscripción. Modelos. Protocolos de Entrega de Notificación.

Unidad VIII
Programación basada en scripts. Conceptos básicos de scripts. Orígenes de la programación script (shell scripts). Desarrollos modernos de lenguajes tipo script. Tipos de lenguajes de scripting. Características generales: Variables, funciones básicas,estructuras de datos avanzadas, procesamiento numérico, interfaces de usuarios gráficas. Introducción al lenguaje Python Script. Declaración de Funciones. Documentación de funciones. Prueba de módulos. Diccionarios. Listas. Tuplas. Definición de variables. Asignación de múltiples valores de una vez. Formato de cadenas. Correspondencia de listas. Unión y división de cadenas. Procesamiento numérico y análisis estadístico.

VII - Plan de Trabajos Prácticos

Unidad I: Introducción al Lenguaje Java
En este práctico se introducen los conceptos iniciales del lenguaje de programación Java. Los ejercicios están orientados a realizar programas simples que impliquen el uso de: clases primitivas, expresiones, sentencias de flujo de control. Ver Laboratorio I.

Unidad II: Desarrollo de Clases en Java
En este práctico se introducen los conceptos necesarios para la creación de clases y de las relaciones posibles entre ellas en el Lenguaje de Programación Java. En este práctico se implementarán en Java los ejercicios del Práctico I y se incorporarán otros nuevos.

Unidad III: Jerarquía de Clases de Java
En este práctico se desarrollarán ejercicios con la utilización de las interfaces de: vector, stack, list, etc. Ver Laboratorio II.

Unidad IV: Manejo de Archivos en Java
En este práctico se desarrollarán ejercicios que utilicen las funciones provistas por Java para la manipulación de archivos.

Unidad V: Mecanismos de Manejo de Excepciones en Java
Los ejercicios desarrollados en el práctico correspondiente a la unidad III serán mejorados introduciendo rutinas de manejo de excepciones, errores, etc.

Unidad VI: AWT y SWING
Ver Laboratorio III.

Unidad VII: Programación Dirigida por Eventos
Ver laboratorio IV.

Unidad VIII: Programación script en Python.
Ver laboratorio V.

Laboratorio I:
Implementación y ejecución de programas cortos usando el lenguaje orientado a objetos Java.

Laboratorio II:
Implementación y ejecución de programas que impliquen el uso de las jerarquías de clases disponibles para el lenguaje orientado a objetos Java.

Laboratorio III:
Entorno de desarrollo en Java (AWT-SWING). Implementación de programas con manejo de interfaces GUI manejada por eventos.

Laboratorio IV:
Desarrollo de programas cortos utilizando la programación dirigida a eventos.

Laboratorio V:
Desarrollo de scripts utilizando el lenguaje Python.

VIII - Regimen de Aprobación

La materia se desarrolla con la modalidad de promoción sin examen final. Existen dos niveles:
a) Regularización solamente: Para regularizar la materia se deberá:
1.- Tener como mínimo un 80% de asistencia a clases prácticas.
2.- Tener los prácticos, solicitados por la cátedra, aprobados.
3.- Aprobar dos parciales o sus respectivas recuperaciones con un mínimo del 60%.
b) Promoción sin examen final: Para regularizar y aprobar la materia se deberá:
1.- Cumplir con los requisitos a.1 y a.2.
2.- Aprobar dos parciales o sus respectivas recuperaciones con un mínimo del 70%.
3.- Aprobar una evaluación integradora global con un mínimo del 70%.
Se otorga dos recuperaciones por cada parcial y práctico de máquina según reglamentación vigente.
Aquellos alumnos que sólo regularicen la materia deberán rendir un examen final, en los turnos establecidos.

IX - Bibliografía Básica

[1] Cachero, Cristina, Ponce de León, Pedro J., Saquete, Estela. "Introducción a la Programación Orientada a Objetos". Universidad de Alicante - Publicación Alicante - 2006.
[2] Y. Daniel Liang, “Introduction to Java Programming, comprehensive”, Prentice Hall; 8 ed, 2010.
[3] Grant Palmer. “Java Event Handling”. Prentice Hall. ISBN 0-13-041802-1. 2001.
[4] David Luckham. “The Power of Events - An Introduction to Complex Event Processing in Distributed Enterprise Systems”. Addison-Wesley. ISBN 0-201-72789-7. 2002.
[5] George S. Fishman. “Discrete-Event Simulation - Modeling, programming, and Analysis”. Springer. ISBN: 0-387-95160-1.
[6] Bertrand Meyer. “Touch of Class: Learning to Program Well with Objects and Contracts”, Springer, 2009.
[7] Bertrand Meyer. “The power of abstraction, reuse and simplicity: an object-oriented library for event-driven design, in Festschrift in Honor of Ole-Johan Dahl eds”. Olaf Owe et al. Springer-Verlag. Lecture Notes in Computer Science 2635.2004.
[8] "The Java Tutorial". http://java.sun.com/docs/books/tutorial/index.html
[9] Herbert Schildt. “Java: A Beginner’s Guide”. Third Edition. McGraw-Hill/Osborne.
[10] "Python Programming Language" - Official Website http://www.python.org/doc/.
[11] Mark Lutz. "Programming Python", O'Reilly Media, Inc.; 3 edition, 2006.
[12] Peter Norton, Alex Samuel, David Aitel, et.al. "Beginning Python" , Wiley Publishing, Inc.2005.
[13] Material provisto por la cátedra.

X - Bibliografia Complementaria

[1]

XI - Resumen de Objetivos

Al finalizar el curso se espera que el alumno sea capaz de:
- Conocer y aplicar los conceptos fundamentales del paradigma de programación orientada a objetos (POO).
- Comprender y aplicar los conceptos fundamentales de la programación dirigida por eventos (PDE).
- Comprender y aplicar los conceptos fundamentales de la programación script.

XII - Resumen del Programa

-Programación Orientada a Objeto. Evolución del concepto de tipo de datos. Tipos de datos abstractos. Ocultamiento de la información y encapsulamiento. Definición de clases. Control de Acceso. Herencia. Subclases. Ejemplos en un lenguaje de POO particular. Herencia simple y múltiple. Tipos de datos elementales y estructurados en POO: Java. Estructuras de control. Polimorfismo y ligadura dinámica. Clases y Métodos abstractos. Ejemplos de un lenguaje POO particular. Paquetes. Interfaces. Excepciones. Entrada-salida. Ambientes de programación. Desarrollo de aplicaciones usando librerías.
- Programación dirigida por eventos (PDE). Desde la programación (secuencial) estructurada a la programación dirigida por eventos. Eventos. Creando y ligando manejadores de eventos. PDE en POO. PDE e interfaces de usuario gráficas.
- Programación basada en scripts. Conceptos básicos de scripts. Desarrollos modernos de lenguajes tipo script. Tipos de lenguajes de scripting. Características generales. Introducción al lenguaje Python Script.

XIII - Imprevistos

Serán resueltos por la cátedra a medida que los mismos vayan surgiendo.

XIV - Otros