DENOMINACIÓN DE LA ACTIVIDAD: 11076 – Programación Orientada a Objetos
TIPO DE ACTIVIDAD ACADÉMICA: Asignatura

CARRERA: Licenciatura en Sistemas de Información
PLAN DE ESTUDIOS: 17.13

DOCENTE RESPONSABLE:
Walter Panessi – Profesor Adjunto

EQUIPO DOCENTE:
Pablo Chale – Profesor Adjunto
Tomas Delvechio - Jefe de Trabajos Prácticos
Santiago Ricci - Ayudante de Primera
Andrés Giordano - Ayudante de Primera
Federico Radeljak - Ayudante de Primera
Nicolás Agustín Onofrio - Ayudante de Segunda

ACTIVIDADES CORRELATIVAS PRECEDENTES:

PARA CURSAR:

(11075) Programación II

PARA APROBAR.

(11075) Programación II

CARGA HORARIA TOTAL:

HORAS SEMANALES: 4 hs

HORAS TOTALES: 64 hs

DISTRIBUCIÓN INTERNA DE LA CARGA HORARIA:

TEÓRICAS: 32 hs (50%)

PRÁCTICAS: 32 hs (50%)

PERÍODO DE VIGENCIA DEL PRESENTE PROGRAMA: 2024-2025

CONTENIDOS MÍNIMOS O DESCRIPTORES

Conceptos sobre resolución de problemas complejos. Fundamentos de Programación Orientada a Objetos. Clases e instancias. Encapsulamiento. Jerarquías de clase. Herencia. Polimorfismo. Diseño de objetos complejos. Lenguajes orientados a objetos. Estructuras de control. Estructura de datos como objetos. Metodologías y técnicas para el desarrollo de aplicaciones.

FUNDAMENTACIÓN, OBJETIVOS, COMPETENCIAS

FUNDAMENTACIÓN:

La complejidad de los sistemas de información a automatizar es cada vez mayor, por lo cual los desarrolladores deben utilizar técnicas cada vez más productivas. Desde esta asignatura, se enfatizará en la construcción de arquitecturas de software modulares, extensibles y reusables, conceptos claves para aplicaciones de gran porte. Se introducirá también al alumno en el uso de un lenguaje de modelado gráfico orientado a objetos (UML) aunque estas técnicas se profundizarán luego en las asignaturas de la disciplina Sistemas de Información donde tratarán temas tanto de análisis como de diseño orientado a objetos. Los ejemplos se podrán dar utilizando lenguajes variados que implementan el paradigma de Programación Orientada a Objetos pero las resoluciones de los prácticos se realizarán solamente en Java®.

OBJETIVOS: (específicos)

- Que el estudiante comprenda la importancia de generar código autodocumentado y correctamente estructurado.
- Que el estudiante se introduzca en técnicas modernas de la programación, en particular en aspectos orientada a objetos, su filosofía y mecanismos de abstracción.
- Que el estudiante puede aplicar razonamientos tendientes a balancear las clases que resuelven los problemas planteados, minimizando las dependencias y maximizando la cohesión.
- Que el estudiante adquiera la habilidad de producir código orientado a objeto partiendo de una descripción de un problema a resolver.

COMPETENCIAS:

Un estudiante aprobado de Programación Orientada a Objetos:

- Conoce y aplica correctamente los conceptos de Modularidad, Herencia, Abstracción, Polimorfismo, Encapsulamiento y Delegación.
- Puede plantear una solución orientada a objetos de un problema del mundo real de complejidad media-baja.

- Produce su código obteniendo un conjunto de clases con sus responsabilidades balanceadas.

CONTENIDOS

UNIDAD 1

La crisis del software. Problemas de las técnicas tradicionales (procedurales). Resolución de problemas complejos. El problema de la extensibilidad, el reuso y el mantenimiento.

UNIDAD 2

Conceptos básicos: Encapsulamiento. Information hiding. Objetos y Programa Orientado a Objetos. Comportamiento de un Objeto. Mensaje y Método. Clasificación: Clases e Instancias. Instanciación. Jerarquías de Clases. Relación isA. Generalización / Especialización. Herencia, Herencia Simple. Clases Abstractas. Hacia mayor genericidad de código: polimorfismo y binding dinámico.

UNIDAD 3

Lenguajes orientados a objetos: variantes. El lenguaje Java. Tipos de Mensajes. Variables de instancia. Utilización de "this" y "super". Método Constructor y Destructor. Concepto de Paquetes. Paquetes Standard

UNIDAD 4

Estructuras de control. Manejo de excepciones. Depuración de programas. Diseño de objetos complejos. Relaciones entre Objetos. Relación de conocimiento. Relación isPartOf. Clases Abstractas. Interfaces. Variables de Clase

UNIDAD 5

Estructuras de datos como Objetos. Objetos contenedores. Colecciones de Objetos. Iteración en colecciones. Implementación de tipos abstractos de datos con objetos.

UNIDAD 6

Aplicación de técnicas orientadas a objetos para la construcción de aplicaciones. Patrones de Diseño Model-View-Controller, Observer y Singleton. Introducción a la persistencia de objetos: ORM, Serialización

METODOLOGÍA DE LA ENSEÑANZA:

El curso se dicta en la modalidad híbrida y es de carácter teórico-práctico. Las clases teóricas, salvo algunas excepciones, se graban en video se ponen a disposición de las y los estudiantes en el aula virtual asignada dentro del espacio de la plataforma institucional. La disponibilidad se realiza en el momento en que es requerida, según la planificación realizada. Las clases

prácticas son 100% presenciales, pudiéndose transmitir, dependiendo de la disponibilidad de herramientas, simultáneamente en alguna plataforma de streaming. Si esto fuera posible, la grabación de las clases también estará disponible para ser consultada en la plataforma. Las clases sincrónicas (presenciales o mediadas por tecnología) son grabadas y, finalizadas las mismas, se ponen los videos a disposición de los estudiantes. Las evaluaciones parciales/recuperatorios se realizan en forma síncrona (presencial) en cada sede. Al comienzo del ciclo se entrega un cronograma con cada una de las actividades que se desarrollará en función de la propuesta pedagógico-didáctica que se está realizando.

Desde el punto de vista teórico, se pone el acento en el paradigma de orientación a objeto y sus fundamentos. Utilizando los conceptos de éste paradigma, los estudiantes deben poder asignar responsabilidades a las clases que identifiquen, de modo que las unidades sean mantenibles, minimizando las dependencias y maximizando la cohesión. Si bien los temas previstos de la asignatura se distribuyen en este programa en unidades temáticas, los conceptos no se brindan en un orden taxativo. A excepción de la primera unidad, que le da fundamento al paradigma, el resto de los conceptos se van agregando lentamente, utilizando definiciones de problemas e instando a los estudiantes a pensar soluciones que conduzcan a descubrirlos, reutilizando, además, los conceptos previamente explicados para reforzar los nuevos y permitiendo verlos desde otra perspectiva, lo que genera una mejor comprensión en forma global. Desde la perspectiva práctica, se definen nuevas situaciones que los estudiantes deben implementar, utilizando un lenguaje de programación orientado a objetos. Finalizando el curso, los estudiantes desarrollan una aplicación seleccionada por el equipo docente y en forma individual. En su desarrollo, los estudiantes deben integrar todos los conceptos aprendidos.

ACTIVIDADES PRÁCTICAS:

El equipo docente ha desarrollado un conjunto de actividades prácticas, de creciente dificultad, que acompaña el desarrollo de la asignatura. Estas actividades se alinean con los conceptos explicados recientemente en forma teórica. Las primeras prácticas, están más orientadas al descubrimiento y utilización de la sintaxis del lenguaje de programación utilizado. Cada práctico plantea problemas que deben ser resueltos utilizando los conceptos explicados en clase como Herencia, Clases Abstractas, Interfaces, Polimorfismo, etc. Algunas prácticas se emplean para forzar el aprendizaje de algunas cuestiones propias del lenguaje como manejo de excepciones, Interfaces Gráficas de Usuario o Persistencia.

En todos los casos, los prácticos son obligatorios (es requisito para regularizar la asignatura la presentación de un porcentaje alto de los mismos) y el equipo docente los corrige generando una devolución de la corrección. Dada la cantidad de estudiantes, no se corrigen todos los ejercicios sino aquellos que se marcan como importantes para el TP.

Para cada unidad teórica, se ha preparado un test de seguimiento que intenta medir el grado de comprensión del tema. La evaluación es multiple – choice. Se realiza y resuelve dentro del aula virtual. La valoración de las respuestas es automática y su devolución es seguida de la realización y envío. La realización de las mismas es, también, obligatoria y forma parte de las condiciones para regularizar la asignatura.

REQUISITOS DE APROBACION Y CRITERIOS DE CALIFICACIÓN:

CONDICIONES PARA PROMOVER (SIN EL REQUISITO DE EXAMEN FINAL)
DE ACUERDO AL ART.23 DEL RÉGIMEN GENERAL DE ESTUDIOS RESHCS-LUJ:0000996-15

a) Tener aprobadas las actividades correlativas al finalizar el turno de examen extraordinario de ese cuatrimestre.
b) Cumplir con un mínimo del 80 % de asistencia
c) Tener realizado el 100% de los test de seguimiento.
d) Aprobar todos los trabajos prácticos previstos en este programa, pudiendo recuperarse hasta un 25% del total por ausencias o aplazos
e) Aprobar el 100% de las evaluaciones previstas con un promedio no inferior a seis (6) puntos sin recuperar ninguna.
f) Aprobar una evaluación integradora de la asignatura con calificación no inferior a siete (7) puntos.

CONDICIONES PARA APROBAR COMO REGULAR (CON REQUISITO DE EXAMEN FINAL)
DE ACUERDO AL ART.24 DEL REGIMEN GENERAL DE ESTUDIOS RESHCS-LUJ:0000996-15

a) estar en condición de regular en las actividades correlativas al momento de su inscripción al cursado de la asignatura.
b) Cumplir con un mínimo del 70 % de asistencia para las actividades
c) Tener realizado el 100% de los test de seguimiento
d) Aprobar todos los Trabajos Prácticos previstos en este programa, pudiendo recuperarse hasta un 40% del total por ausencias o aplazos
e) Aprobar el 100% de las evaluaciones previstas con un promedio no inferior a cuatro (4) puntos, pudiendo recuperar el 50% de las mismas. Cada evaluación solo podrá recuperarse en una oportunidad.

EXÁMENES PARA ESTUDIANTES EN CONDICIÓN DE LIBRES

Para aquellos estudiantes que, habiéndose inscriptos oportunamente en la presente actividad hayan quedado en condición de libres por aplicación de los artículos 29 o 32 del Régimen General de Estudios, podrán rendir en tal condición la presente actividad, siempre que presenten 15 días antes de la fecha del examen, la guía de trabajos prácticos completa.

La misma puede ser solicitada al correo electrónico wpanessi@unlu.edu.ar Para poder presentarse, esta guía debe estar aprobada.

BIBLIOGRAFÍA

Bibliografía Obligatoria:

1. DEITEL, "COMO PROGRAMAR EN JAVA" - PRENTICE HALL (PEARSON) - Séptima Edición – 2008 - ISBN 9789702611905
2. BRUCE ECKEL "PIENSA EN JAVA" – PRENTICE HALL – 2002 - ISBN 9788489660342
3. ADAM DROZDEK, "ESTRUCTURAS DE DATOS Y ALGORITMOS EN JAVA" - THOMSON - Segunda Edición – 2007 - ISBN 9789706866110
4 – LARMAN, Craig - "UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado" - Segunda Edición – PEARSON EDUCATION – 2002

Bibliografía de Consulta:

1. BOOCH GRADY, "ANALISIS Y DISEÑO ORIENTADO A OBJETOS CON APLICACIONES" ADDISON-WESLEY IBEROA – 1996 - Edición Número 2 ISBN 9684443528
2. WIRFS-BROCK, Rebecca; WILKENSON Brian (Contributor), WIENER, Lauren. Textbook Binding "DESIGNING OBJECT-ORIENTED SOFTWARE" - Prentice Hall PTR – 1991 - ISBN 0136298257
3. RUMBAUGH JAMES, BLAHA MICHAEL , EDDY FREDERICK , LORENSEN WILLIAM , PREMERLANI WILLIAM – "MODELADO Y DISEÑO ORIENTADO OBJETOS" - PRENTICE-HALL – 1996 - ISBN 0132406985
4. BOOCH GRADY, JACOBSON IVAR , RUMBAUGH JAMES, "EL LENGUAJE UNIFICADO DE MODELADO" - ADDISON-WESLEY IBEROA – 2000 - ISBN 8478290281
5. OMG, "OMG UNIFIED MODELING LANGUAGE SPECIFICATION". At http://www.omg.org/spec/UML/2.4.1/

DISPOSICIÓN CD

Firmado digitalmente por PANESSI Walter Fabián
Fecha: 2024.08.10 19:55:25 -03'00'