473/18 18 ecados

Aprobado por resolución N°113 del CFI de fecha 04.07.2017

FACULTAD DE INGENIERIA

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Programa de Programación 4

1. NOMBRE DE LA UNIDAD CURRICULAR

Programación 4

2. CRÉDITOS

15 créditos

3. OBJETIVOS DE LA UNIDAD CURRICULAR

Generales:

- Introducir elementos necesarios para la construcción de sistemas de software medianos y grandes.
- Presentar y aplicar para ello conceptos de la orientación a objetos.

Particulares:

- Dar a conocer herramientas conceptuales para el análisis y diseño de sistemas orientados a objetos.
- Presentar una metodología básica para el uso de dichas herramientas.
- Dar a conocer un lenguaje de programación que permita expresar los conceptos involucrados en la orientación a objetos.
- Adquirir experiencia en ese lenguaje para poner en práctica los conceptos anteriores.

4. METODOLOGÍA DE ENSEÑANZA

La enseñanza se realiza mediante clases expositivas de teórico y práctico. Además existe un componente de laboratorio donde el estudiante debe resolver problemas en forma grupal, aplicando los conocimientos de la unidad curricular. El detalle de horas de cada componente se muestra en la siguiente tabla.

	Asistencia	Estudio/ Preparación	Total	Descripción	Obligatoria
Teórico	40	20	60	Dos clases semanales de dos horas	No
Práctico	40	30	70	Dos clases semanales de dos horas	No
Laboratorio	4	60	64	Una clase semanal de 20 minutos	Sí
Evaluación	4	20	24	Defensa de laboratorio y parcial	Sí
Otros			7	Participación remota en sitio web	No
Total			**225**		

5. TEMARIO

1. Introducción a la orientación a objetos: Etapas del desarrollo de software, arquitectura. Objetos, clases, asociaciones, herencia, polimorfismo, despacho. Lenguaje de modelado.
2. Análisis orientado a objetos: Modelado del dominio, abstracción y formalización. Estructura y comportamiento. Especificación en lenguaje de modelado.
3. Diseño orientado a objetos: Modelado de la solución, asignación de responsabilidades. Estructura y comportamiento. Especificación en lenguaje de modelado.
4. Implementación orientada a objetos: Clases, constructores y destructores. Sobrecarga de funciones y operadores. Herencia, polimorfismo, despacho dinámico. El pasaje del diseño a la implementación. Manejo de objetos y colecciones. Implementación en lenguaje de programación.

6. BIBLIOGRAFÍA

Tema	Básica	Complementaria
Introducción a la orientación a objetos	(1), (2), (5)	(7)
Análisis orientado a objetos	(1), (2), (5), (6)	(10)
Diseño orientado a objetos	(1), (2), (3), (5), (6)	(10)
Implementación orientada a objetos	(1), (4), (5)	(8), (9), (11), (12), (13)

Para cada item de la bibliografía se indica si está disponible en las bibliotecas de Facultad de Ingeniería (Fing) y del Instituto de Computación (InCo).

6.1 Básica

1. Larman, C (2004). Applying UML and patterns. Prentice Hall. ISBN-13: 978-0131489066 ; ISBN-10: 0131489062. (Fing, InCo)
2. Fowler, M (2003) UML Distilled. Addison Wesley. ISBN-13: 978-0321193681 ; ISBN-10: 0321193687 .
3. Gamma, E; Helm, R; Johnson, R; Vlissides, J (1994) Design Patterns. Addison-Wesley Professional. ISBN-10: 0201633612 ; ISBN-13: 978-0201633610.
4. Deitel, HM; Deitel PJ (2004) Cómo programar en C/C++ y Java. Prentice Hall. ISBN-10: 9702605318 ; ISBN-13: 978-9702605317.
5. Docentes de la unidad curricular. Diapositivas de teórico. https://eva.fing.edu.uy/mod/page/view.php?id=24643

6.2 Complementaria

6. Booch, G; Rumbaugh, J; Jacobson I (2005) The Unified Modeling Language User Guide. Addison Wesley. ISBN 978-032-126-797-9. (Fing, InCo)
7. Krutchen, P (2003) The Rational Unified Process: An Introduction. Addison Wesley. ISBN 0-321-19770-4. (Fing)
8. Stroustrup, B (2013) The C++ Programming Language. Addison-Wesley Professional. ISBN-10: 0321958322 ; ISBN-13: 978-0321958327. (Fing, InCo)
9. Josuttis, NM (2012) The C++ Standard Library. Addison-Wesley Professional. ISBN-10: 0321623215; ISBN-13: 978-0321623218.
10. Object Management Group. Unified Modeling Language. http://www.omg.org/spec/UML/
11. GCC, the GNU Compiler Collection. https://gcc.gnu.org/
12. Code::Blocks. http://www.codeblocks.org/
13. GitHub. https://github.com/

7: CONOCIMIENTOS PREVIOS EXIGIDOS Y RECOMENDADOS

7.1 Conocimientos Previos Exigidos: Teoría de conjuntos y lógica de predicados. Programación estructurada. Tipos abstractos de datos y modularización. Manejo de memoria y estructuras de datos.

7.2 Conocimientos Previos Recomendados: Lenguajes de programación estructurada. Herramientas de desarrollo (compilador, debugger, entorno de programación) y de gestión de proyectos de software (repositorios de código, trabajo grupal).

ANEXO A
Para todas las Carreras

A1) INSTITUTO

Instituto de Computación

A2) CRONOGRAMA TENTATIVO

En todas las semanas (excepto las dos últimas) se dictan cuatro horas de clases téoricas en los temas especificados. Los cronogramas de práctico y laboratorio acompañan los temas de teórico, con un leve desfasaje para permitir la adquisición de los conocimientos necesarios.

Semana 1	Introducción al curso y desarrollo orientado a objetos
Semana 2	Conceptos básicos de orientación a objetos
Semana 3	Conceptos básicos de orientación a objetos
Semana 4	Requerimientos e introducción al análisis, modelado de dominio
Semana 5	Análisis, comportamiento
Semana 6	Introducción al diseño
Semana 7	Diseño, comportamiento
Semana 8	Diseño, asignación de responsabilidades
Semana 9	Diseño, aspectos avanzados
Semana 10	Diseño, aspectos avanzados
Semana 11	Implementación, generación de código y manejo de objetos
Semana 12	Implementación, colecciones de objetos
Semana 13	Implementación, aspectos avanzados
Semana 14	(Finalización de laboratorio)
Semana 15	(Evaluación de laboratorio)

A3) MODALIDAD DEL CURSO Y PROCEDIMIENTO DE EVALUACIÓN

La unidad curricular se evalúa por medio de una prueba individual final (parcial) y por los trabajos de laboratorio. De los resultados obtenidos por el estudiante surgen tres categorías:

a) Exoneración del examen.
b) Suficiencia en el curso, que habilita a rendir el examen.
c) Insuficiencia en el curso, por lo cual se reprueba, debiendo reinscribirse en el mismo.

Para determinar la categoría se considera el puntaje final, teniendo en cuenta:

1) Las franjas

Exoneración: puntaje total mayor o igual al 60%
Suficiencia: puntaje total entre 25 y 59%
Insuficiencia: puntaje total menor al 25%

2) Los laboratorios

Las clases de laboratorio son de asistencia obligatoria. El laboratorio consta de varias entregas, cuya calificación se combina para calcular una calificación individual de laboratorio. El puntaje obtenido en el laboratorio será eliminatorio (franja Insuficiencia) para aquellos estudiantes que no obtengan al menos el nivel de suficiencia.

3) Puntaje total

Se calculará teniendo en cuenta los puntajes del laboratorio (para aquellos estudiantes que alcanzaron el nivel de suficiencia) y de la prueba final.

A4) CALIDAD DE LIBRE

La unidad curricular no adhiere a la resolución de calidad de libre.

A5) CUPOS DE LA UNIDAD CURRICULAR

No tiene.

ANEXO B para la carrera Ingeniería en Computación (plan 97)

B1) ÁREA DE FORMACIÓN

Programación.

B2) UNIDADES CURRICULARES PREVIAS

Para el Curso:

Curso de Programación 3 y
Exámenes de Cálculo diferencial e Integral en una variable (o Cálculo 1) y
Geometría y Álgebra Líneal 1 y
Programación 2 y
Matemática Discreta 1.

Para el Examen:

Curso aprobado de Programación 4.

Aprobado por resolución N°113 del CFI de fecha 04.07.2017

ANEXO B para la carrera Ingeniería en Computación (plan 87)

B1) ÁREA DE FORMACIÓN

No corresponde

B2) UNIDADES CURRICULARES PREVIAS

Para el Curso: Exámenes de Análisis Matemático I y
Álgebra Líneal y
Lógica y
Curso de Taller II

Para el Examen: Curso aprobado de Programación 4.

Aprobado por resolución Nº113 del CFI de fecha 04.07.2017

ANEXO B para la carrera Ingeniería Eléctrica

B1) ÁREA DE FORMACIÓN

Informática

B2) UNIDADES CURRICULARES PREVIAS

Curso: exámenes de:

- Diseño Lógico
- Programación 1
- Calculo diferencial e integral en una variable
- Geometría y Algebra Lineal 1

Examen: curso de Programación 4.