Facultad de Ingeniería
Sazie 2325, 3er Piso, Santiago
+56 2 661 8256

SYLLABUS de la Asignatura

Programación – INF1201

1. Descripción de la asignatura

El estudiante aprenderá a analizar un problema y plantear una solución racionalmente, utilizando programación orientada a objetos, aplicando los fundamentos de objetos y clases mediante un ambiente IDE que permita el desarrollo y depuración de código del lenguaje. Además, será capaz de describir y entender el proceso de compilación y ejecución.

2. Prerrequisitos, Co-requisitos y Horas Pedagógicas

Prerequisito: no tiene.
Co-requisito: no tiene.
Horas:
Teórico: 4 hrs.
Laboratorio: 2 hrs.

3. Objetivos

Objetivo General

Conocer, comprender, aplicar y analizar problemas de ingeniería, utilizando programación estructurada.

Objetivos Específicos

- Conocer los fundamentos de la computación e informatica.
- Analizar y solucionar problemas sencillos de ingeniería a nivel conceptual.
- Conocer y comprender los fundamentos de la programación estructurada.
- Desarrollar aplicaciones orientadas a objetos utilizando un IDE.

4. Aprendizajes Esperados

Al finalizar la asignatura el estudiante debe:

- Desarrollar el pensamiento algorítmico para resolver problemas de ingeniería mediante el uso de un lenguaje de programación.
- Conocer la lógica y metodología para el desarrollo de aplicaciones utilizando un lenguaje de programación estructurada.
- Analizar problemas identificando sus componentes relevantes.
- Resolver problemas mediante la descomposición de estos en otros más pequeños.
- Resolver problemas mediante el desarrollo de programas computacionales.
- Aplicar las instrucciones básicas de un lenguaje de programación estructurada: secuencia, sentencias de selección y sentencias de repetición.
- Diseñar programas modulares.

5. Sistema de Evaluación de la Asignatura

La nota de presentación a examen se calculará como sigue:

Control 1	= 30%
Control 2	= 30%
Laboratorio	= 40%

Si la nota de presentación a examen es igual o superior a 5.0, el alumno o alumna podrá eximirse de la rendición del examen de la asignatura, siempre y cuando no presente notas parciales bajo 4,0.

Los alumnos que no se eximan calcularán su calificación final según lo siguiente:

Nota de presentación * 0.7 + Nota de examen * 0.3

Las fechas de evaluaciones serán informadas oportunamente.

La asistencia requerida para aprobar el curso es 75%, de forma contraria se reprobará el curso con nota final igual a:

MÍN(nota final, 3.9)

Finalmente la aprobación del curso es por separado para controles y laboratorio. O sea, se requiere nota igual o superior a 4.0 tanto en el promedio de controles como en el promedio de laboratorio para poder aprobar el curso. En caso contrario la nota de presentación y por ende la nota de examen será:

MÍN(promedio controles, promedio laboratorio)

6. Actividades del Curso

Cátedra

El alumno o alumna deberá asistir a clases teóricas y rendir dos controles, las cuales serán escritas y de acuerdo a los contenidos que se hayan visto en clases, los cuales, por lo general, serán acumulativos, esto significa que para la segunda prueba será necesario utilizar lo aprendido para la primera.

Las clases serán expositivas, donde se explicarán conceptos y adicionalmente se mostrarán diversos ejemplos, tanto conceptuales como prácticos para apoyar los contenidos presentados.

Laboratorio

Corresponde a la nota por participación y tareas en los laboratorios asociados al curso. El laboratorio se abordará como una instancia donde los trabajos irán siendo acumulativos, por lo cual los alumnos deberán utilizar lo aprendido en los laboratorios anteriores.

Los contenidos a reforzar con los ejercicios de los laboratorios son los mismos que los de la cátedra, y deberán ir al mismo nivel.

El laboratorio es para programas sencillos, de ejecución en una terminal, no se espera que el alumno desarrolle una aplicación gráfica durante el transcurso del mismo.

7. Calendario de contenidos y actividades del curso

Unidad de aprendizaje	Aprendizajes Esperados	Clases	Estructura de Contenidos	Observaciones
Introducción	Comprender aspectos básicos de los lenguajes de programación, su historia y evolución	1	- Computación vs informática - Datos vs información - Arquitectura de von Neumann - Lenguajes de programación	
		2-3	- Definición de problemas - Seudo-código - Modelos de programación - Licencias de software	
Programación estructurada	Estudiar conceptos relacionados con el paradigma de programación estructurada utilizando las estructuras de secuencias, selección e iteración.	4	- Intérprete - Expresiones - Tipos de datos - Entrada y salida estándar	
		5-6	Punteros	
		7	- Estructuras condicionales	
		8-9	- Estructuras de repetición	
Funciones y módulos	Realizar funciones que permitan encapsular código, y de la misma forma agruparlas en módulos que sean reutilizables	10-11	- Uso y definición de funciones - Creación y uso de módulos	
		12-13	- Recursividad y ordenamiento	
		14		Control 1
		15		Entrega control 1
Programación orientada a objetos	Estudiar conceptos relacionados con el paradigma de programación orientada a objetos, creación de clases y propiedades de las mismas	16	- Clases, atributos y métodos	
		17-18	- Herencia y polimorfismo	
Archivos	Trabajar con la entrada y salida de datos hacia ficheros	19-21	- Leer archivos - Escribir archivos	
Estructuras de datos simples	Usar diferentes tipos de estructuras de datos simples y como permiten colaborar con la programación	22	- Estructuras de datos básicas (como arreglos)	
		23	- Definición de nuevas estructuras de datos.	

		24	Arreglos de estructuras	
		25-26	- Arreglos bidemensionales (matrices)	
Procesamiento de texto	Procesar texto, realizando diferentes operaciones sobre el mismo	27	- Reemplazar - Búsqueda en cadenas - Unir y divir - Expresiones regulares	
		28		Control 2
		29		Entrega Control 2
		30		Repaso examen

8. Bibliografía del Curso

Bibliografía Obligatoria

- Kernighan, B., Ritchie, D. (1988). *The C Programming Language, 2nd Edition*.
- Ceder, V. (2010). The Quick Python Book. Londres: Manning.
- Downey, A. (2009). Python for Software Design. Londres: Cambridge.
- Campbell, J., Gries, P., Montojo, J., Wilson, G. (2009). Practical Programming: An Introduction to Computer Science Using Python. Estados Unidos: The Pragmatic Bookshelf.
- Lutz, M. (2009). Learning Python. Estados Unidos: O'Reilly Media.

Bibliografía Extra

- Joyanes, L. (2003). Fundamentos de programación: Algoritmos, estructura de datos y objetos. Madrid: McGraw-Hill.
- Joyanes L. Fundamentos de Programación - Libro de Problemas. Mc GrawHill.
- Lemay, L. Perkins, C. (1996). Aprendiendo Java en 21 días. Prentice Hall.
- Deitel, H., Deitel, P. Como programar en C++. Pearson educación.
- Deitel, H., Deitel, P. Como programar en C#. Pearson educación.

Recursos en línea

- Curso de Programación en USM: http://progra.usm.cl
- Documentación oficial Python: http://docs.python.org
- Python no muerde: http://nomuerde.netmanagers.com.ar
- Compilador online: http://profesores.ing.unab.cl/~delaf/ide