UNIVERSIDAD ADOLFO IBÁÑEZ

Syllabus de asignatura TICS200 Lenguajes y Paradigmas de Programación

Unidad académica	Pregrado		
Carrera o programa	Ingeniería Civil Informática		
Año	2019	Semestre	2
Profesor	Gonzalo Huerta Cánepa (Viña) Felipe Aguilera Valenzuela (Santiago)	Email	gonzalo.huerta@uai.cl felipe@aguilera.cl
		Horario de atención	Ma 11:30 – 14:10 (Viña) Ma 15:00 - 17:40 (Santiago)
Ayudante	-	Email	-
Créditos SCT-Chile	6	Total horas	180

Horas de Docencia Directa			Horas de Trabajo Autónomo
Cátedra	Laboratorio	Ayudantía	
45	15	0	120

Tipo de Asignatura	Plan Común
Línea curricular/ Área	Informática
Pre-requisitos	TICS100 (Programación), nociones de algorítmica y estructura de datos.
Descripción de la asignatura	Aprender a desarrollar en diferentes lenguajes de programación es crucial para el desarrollador global. Saber programar cualquier lenguaje proporciona una herramienta poderosa (y necesaria) para desenvolverse en el mercado actual, permitiendo a los desarrolladores de software seleccionar el lenguaje que mejor se adapta a las necesidades de un proyecto. Esto se ve afianzado por el desarrollo de lenguajes que se ejecutan sobre una plataforma común. Para llevar a cabo el objetivo de aprendizaje, el curso se organiza en 3 partes: una sección introductoria donde los estudiantes aprenden sobre la historia de los lenguajes de programación, los diferentes paradigmas y un lenguaje de programación representativo de cada una de los paradigmas, una sección de aplicación donde se estudian casos relacionados con los diferentes paradigmas, y una sección de aplicación de lo aprendido, donde los estudiantes desarrollan una aplicación final (proyecto).
Competencias del egresado	i), j), k)
Resultados de Aprendizaje	Al final del curso se espera que el alumno entienda los conceptos más importantes de las etapas de análisis, diseño y programación en los distintos paradigmas, y sea capaz de seleccionar los más adecuados y aplicarlos correctamente en las distintas etapas del desarrollo de software. En particular, el estudiante podrá: - reconocer los diferentes paradigmas de programación - determinar en qué situaciones el uso de uno u otro ayuda a la implementación de software para un problema determinado. - modelar una solución a problemas simples descritos en lenguaje natural mediante la encapsulación en componentes reutilizables que permiten minimizar la duplicidad de código y mejorar la depuración de este - desarrollar aplicaciones usando herramientas profesionales de desarrollo de software.

Estrategias de enseñanza y aprendizaje

Los contenidos formales y las actitudes frente al tema se entregarán a través de clases expositivas. Las habilidades específicas se mostrarán en las clases y se practicarán mediante ejercicios efectuados en clases y tareas. Se considera tener asignaciones, para reforzar los conceptos en los distintos paradigmas de programación.

Procedimientos de Evaluación de aprendizajes

El curso contará con 2 sesiones semanales de cátedra, con énfasis en trabajo práctico donde el profesor de cada sección junto con los ayudantes actuarán como facilitador del aprendizaje del alumno. Las sesiones están diseñadas de manera que promuevan el aprendizaje activo de los alumnos. Por ejemplo, mediante el planteo de problemas que los alumnos deben resolver en clases o la discusión crítica de algoritmos y soluciones propuestas por parte de los alumnos. Cada clase se asignará un trabajo a realizar, el cuál será evaluado mediante una discusión al comienzo de la clase siguiente.

El curso contará con los siguientes mecanismos de evaluación:

- 2 pruebas: Las pruebas medirán a lo largo del semestre la capacidad del alumno para leer e interpretar un programa computacional, además de su capacidad para escribir programas simples.
- Actividades en aula: Cada concepto enseñado en Auditorio o aula, se evaluará mediante desafíos que los alumnos deben resolver en clases.
- 1 proyecto: El proyecto consiste en el desarrollo de una aplicación que resuelva un problema de la vida real usando las herramientas entregadas durante el curso.
- 1 examen final: el examen mide de manera acumulativa las capacidades del alumno para leer, interpretar y escribir programas computacionales.

Unidades de la asignatura (máximo 1 plana)

A. Paradigmas de programación (9 semanas)

1. Historia y conceptos de programación
2. El paradigma procedural.
3. Paradigma orientada al objeto.
4. Paradigma funcional
5. Paradigma lógico

B. Aplicación de Paradigmas de Programación (2 semanas)

6. Estudio de casos
7. Selección de proyectos y paradigmas a utilizar

C. Proyecto semestral

Reglamento

Mecanismo de evaluación:

- 1ª Prueba Oficial (P1): 30%
- 2ª Prueba Oficial (P2): 30%
- Asignaciones (A): 30%
- Libre disposición del profesor (LP): 10%

La Nota Final (NF) se calculará de la siguiente forma:

- Se calculará una Nota presentación a Examen: NPE = 0,3 P1 + 0,3 P2 + 0,3 A + 0,10 LP.
- Si NPE >= 5,0 y P1, P2, A y LP >= 4,0 y NPE dentro del primer decil de NPEs del curso, el profesor podría eximir al alumno de rendir examen y, entonces: NF = NPE
- Si el alumno debe rendir examen y la nota obtenida en el examen (NE) es mayor o igual a 3,0, entonces NF = 0,30 NE + 0,70 NPE. Si NE es menor a 3,0, entonces NF = NE.

Respecto a la asistencia:

- Se requiere que el alumno asista mínimo al 70% de las sesiones.
- La inasistencia a alguna evaluación se calificará con un 1,0. La justificación dentro de los 3 días hábiles de ocurrida la inasistencia y aceptada por la Facultad dará derecho a recuperar dicha evaluación según lo que se estipula a continuación:
- Inasistencia justificada a una prueba oficial: podrá rendir el examen final y la evaluación de este equivaldrá a la evaluación de la prueba no rendida y simultáneamente a la del examen mismo. Nótese que al rendir examen, eso significa que debe obtener una nota mínima de 3,0 en el examen para poder aprobar el curso.
- Inasistencia justificada a dos pruebas oficiales: aplica el punto anterior para una de las pruebas oficiales. Además, deberá rendir una prueba recuperativa que sustituirá a la prueba oficial 2.

Notas adicionales sobre el curso

1. Webcursos (http://webcursos.uai.cl) es parte integral del curso. Es su responsabilidad visitar periódicamente el sitio con el fin de informarse de las novedades y comunicaciones relacionadas con este.
2. Comunicación con el profesor fuera de clases. Utilice las direcciones de correo electrónico que aparecen en Webcursos. Si lo estima necesario, Ud. puede solicitar por correo electrónico una reunión con su profesor.
3. Comportamiento en clases:
    a. Llegada atrasada a clases: Atraso máximo de 10 minutos. Por favor no entre a la sala si su atraso excede 10 minutos.
    b. Apague su teléfono celular.
    c. Fomente un ambiente de respeto a sus compañeros, al profesor, y conducente al aprendizaje.
    d. En particular evite salidas y entradas de la sala de clases. Es una falta de respeto grave leer diarios, trabajar en puzzles, trabajar en tareas de otros cursos, etc. En caso de una falta reiterada en clases, el profesor dispone de la facultad de aplicar un Fail al alumno.
2. Asistencia a ayudantías es obligatoria. El ayudante pasará lista todas las clases. Es su responsabilidad mantenerse al tanto de información académica o administrativa que el profesor o ayudante comuniquen a los alumnos.

Recursos para el Aprendizaje (Bibliografía)

1. Structure and Interpretation of Computer Programs, by Abelson, Sussman, and Sussman. MIT Press (disponible online en https://mitpress.mit.edu/sicp/)
2. Programming Language Pragmatics, Fourth Edition 4th Edition, by Michael L. Scott. Morgan Kaufmann (2015)

Otra bibliografía para el curso, compuesta mayormente por material en video y páginas web están disponibles online en el sistema Webcursos.

UNIVERSIDAD ADOLFO IBÁÑEZ

Clase a clase

Fecha	Contenidos	Actividades de aprendizaje	Recursos
Semana 1	Historia y conceptos de programación	Clase expositiva de introducción usando la historia de los lenguajes de programación como base.	Pizarra Cuaderno de apuntes
Semana 2	Historia y conceptos de programación	Clase expositiva y discusión de diferencias entre los paradigmas en forma general.	Pizarra Cuaderno de apuntes
Semana 3	Paradigma procedural	Clase introductoria a lenguajes procedurales, usando C o PHP como ejemplo, dado que los alumnos vieron uno de esos lenguajes.	Pizarra Sitio web: repl.it Cuaderno de apuntes
Semana 4	Paradigma orientado al objeto	Clase expositiva que muestra la transición entre los lenguajes procedurales y los orientados al objeto.	Pizarra Cuaderno de apuntes
Semana 5	Paradigma orientado al objeto	Clase práctica con ejemplos en clase.	Pizarra Sitio web: repl.it Cuaderno de apuntes
Semana 6	Paradigma funcional	Clase expositiva que muestra la transición entre los lenguajes procedurales y los funcionales.	Pizarra Cuaderno de apuntes
Semana 7	Paradigma funcional	Clase práctica con ejemplos en clase.	Pizarra Sitio web: repl.it Cuaderno de apuntes
Semana 8	Paradigma lógico	Clase expositiva que muestra la transición entre los lenguajes procedurales y los lógicos.	Pizarra Cuaderno de apuntes
Semana 9	Paradigma lógico	Clase práctica con ejemplos en clase.	Pizarra Sitio web: repl.it Cuaderno de apuntes

Semana 10	Prueba oficial nº1	-	-
Semana 11	Estudios de casos	Clase expositiva.	Pizarra Web Cuaderno de apuntes
Semana 12	Estudios de casos	Clase expositiva.	Pizarra Web Cuaderno de apuntes
Semana 13	Selección de proyectos y paradigmas a utilizar	Clase de discusión sobre proyectos	-
Semana 14	Prueba oficial nº2	-	-
Semana 15	Proyecto semestral	Tutorías y mentorías de apoyo al desarrollo de proyectos	-
Semana 16	Prueba Oficial Nº 2	Tutorías y mentorías de apoyo al desarrollo de proyectos	-

Competencias del egresado				
Asociar los objetivos y metodología que en su asignatura serán abarcados (no todo curso despliega todos los objetivos) y a su vez enlazar la metodología con una métrica p				
Objetivos		Metodología		
Competencias y Habilidades Objetivo General	**Resultados de aprendizaje** Objetivo Específico	**Contenidos** Indicar los contenidos (se recomienda referir sólo el nivel o subnivel, ej 1.2, 3.1) que se abordarán en las sesiones, ya sean de carácter conceptual, procedimental y/o actitudinal	**Actividades y Recursos** Indicar estrategia de enseñanza y aprendizaje. Describir actividades y recursos a ocupar por los estudiantes: pizarra, software, web, etc.	**Métrica** Acción que un estudi demostrar para apro Indicar procedimiento trabajo, prueba, inter
(a) Aplicar conocimiento de matemáticas, ciencia e ingeniería	Modificación de componentes de sistemas operativos. Se requiere aplicar conocimientos matemáticos para calcular la mejor solución.	3, 4, 5, 6,y 7	Dado un sistema operativo simplificado (XV6), el alumno deberá modificar sus componentes de forma tal que funcionen de otra manera, por ejemplo priorizar según tiempo de uso.	Tareas Prueba
(c) Diseñar un sistema, componente o proceso que cumpla con las necesidades requeridas, considerando restricciones realistas	Modificación de componentes de sistemas operativos dadas restricciones de uso.	3, 4, 5, 6,y 7	Dado un sistema operativo simplificado (XV6), el alumno deberá modificar sus componentes de forma tal que funcionen de otra manera, por ejemplo priorizar según tiempo de uso.	Tareas
(h) Comprender el impacto de las soluciones de ingeniería en un contexto global, económico, ambiental y social	Comprender el impacto que han tenido las tecnologías TI en las organizaciones y en la sociedad, y su impacto en el modo en que ellas operan. ¿Por qué un sistema operativo es clave?.	1 y 2	Sala de clase	Prueba
(i) Reconocer la necesidad y la habilidad para comprometerse en un aprendizaje permanente	Inculcar al alumno capacidades de abstracción y pensamiento algorítmico a través de un proceso de aprendizaje continuo y permanente (programación se aprende programando)	1 y 2	Sala de clase	Prueba
(j) Conocer temas contemporáneos	Trabajar en ambientes de sistemas operativos distribuidos	9 y 10	Taller usando Amazon AWS o Microsoft Azure	Prueba
(k) Usar técnicas, habilidades y herramientas modernas de ingeniería necesarias para la práctica ingenieril	Modificación de componentes de sistemas operativos. Se requiere programar, herramienta base en las ingenierías de hoy.	3, 4, 5, 6,y 7	Dado un sistema operativo simplificado (XV6), el alumno deberá modificar sus componentes de forma tal que funcionen de otra manera, por ejemplo priorizar según tiempo de uso.	Tareas