UNIVERSIDAD ADOLFO IBÁÑEZ

Syllabus TICS-100 Programación

Unidad académica	Pregrado		
Carrera o programa	Plan Común		
Año	2021	Semestre	2
Profesor	Rafael Cereceda Martínez	Email	rafael.cereceda@uai.cl
		Horario de atención	Bajo demanda. Se solicita por correo electrónico.
Ayudantes	Por definir	Email	
Créditos SCT-Chile	6	Total horas	180

Horas de Docencia Directa			Horas de Trabajo Autónomo
Cátedra	Laboratorio	Ayudantía	
45	0	23	112

Tipo de Asignatura	Plan Común
Línea curricular/ Área	Informática
Pre-requisitos	N/A
Descripción de la asignatura	La asignatura se enfoca en el desarrollo del pensamiento algorítmico como principal eje articulador para la resolución y modelación de problemas. El pensamiento algorítmico es la habilidad de **identificar de manera precisa la secuencia completa de pasos** que es necesario llevar a cabo para resolver un problema.
Competencias del egresado	(a), (c) y (e)
Resultados de Aprendizaje	Al final del curso, los estudiantes deben ser capaces de: 1. Escribir código claro, robusto y eficiente en Python usando: - Instrucciones secuenciales, condicionales y de ciclo. - Cadenas de texto, listas, tuplas, conjuntos y diccionarios. - Interacción con usuario 2. Desarrollar programas para resolver eficazmente tareas mediante: - El uso de diseño modular en la construcción del programa. - Creación y escritura proactiva de casos de prueba para probar y depurar código - Diseñar y escribir un programa de tamaño medio (500-1500 líneas) en Python de manera independiente 3. Aplicar habilidades de resolución de problemas computacionales a nuevos problemas, especialmente en la disciplina académica del estudiante

Estrategias de enseñanza y aprendizaje

El curso está diseñado en dos etapas. En la primera fase busca el desarrollo del proceso de abstracción en la resolución de problemas empleando el lenguaje natural. Ello habilita al alumno a desarrollar un pensamiento más estructurado y claro, identificando condiciones de borde y condiciones repetitivas. Este proceso requiere una inversión de tiempo de discusión y análisis para definir las fases que existen en la resolución de un problema. En la segunda fase del curso se busca la transferencia de conocimiento a un lenguaje de programación formal (Python) que permite finalmente la ejecución en el computador de un problema acotado, definido, y estructurado.

Procedimientos de Evaluación de aprendizajes

El curso considera diferentes mecanismos de evaluación: tareas, controles, pruebas, y proyecto, que se detallan a continuación:

(1) *tareas*: Las tareas están compuestas de dos etapas: una asignación preliminar que consiste en completar un conjunto de ejercicios sencillos para comprobar su comprensión de las notas, y luego de una tarea propiamente tal. La asignación preliminar realizará en clases mediante una pregunta durante la cátedra. La tarea debe hacerse en forma individual, a menos que lo contrario sea explícitamente mencionado en el enunciado de la tarea.

(2) *controles*: Los controles se realizarán una vez a la semana, en el horario de ayudantía. Generalmente, estos cubren material de la semana y la tarea anterior. Los controles están diseñados para ser intensivos en el tiempo, para probar la fluidez y demostrar dónde se necesita un estudio adicional.

(3) *pruebas*: miden acumulativamente el aprendizaje del alumno en clases durante el semestre. Habrá tres pruebas en el transcurso del semestre, cuyas fechas serán informadas con anticipación. Estos se enfocan más en pensamiento algorítmico en lugar de fluidez, que es buscada con los controles.

(4) *proyecto:* se evaluará el desarrollo de un proyecto en equipo con uso de módulos de Python que permitan realizar aplicaciones más sofisticadas. Algunos ejemplos son: Django, Pygame, Arcade, Matplotllib, entre otros.

Unidades de la asignatura (máximo 1 plana)

1. Introducción a la resolución de problemas
   - 1.1. Análisis formal de un problema (condiciones borde, ciclos)
   - 1.2. Descripción de un problema en sub-etapas a través de un lenguaje natural
   - 1.3. Técnicas de resolución de un problema
   - 1.4. Hour Of Code

2. Introducción al lenguaje de programación Python
   - 2.1. Introducción a Python
   - 2.2. Trabajando con Python: herramientas
   - 2.3. Noción y manejo de variables
   - 2.4. Uso de operadores lógicos y aritméticos
   - 2.5. Manejo de condiciones (IF-ELSE)
   - 2.6. Introducción al uso de funciones estándar de Python (print, input, entre otras)
   - 2.7. Tipos de datos avanzados: listas, tuplas, sets, diccionarios
   - 2.8. Manejo de ciclos y condicionales anidados
   - 2.9. Uso de módulos de terceros en Python (random, math, numpy, pandas)
   - 2.10. Definición de funciones propias
   - 2.11. Introducción a orientación a objetos en Python

3. Aplicaciones en Python
   - 3.1. Desarrollo de proyecto en Python

Reglamento

Participación:
Las clases poseen actividades interactivas que ayudan al aprendizaje del alumno. Estas actividades requerirán el uso de una computadora portátil o teléfono. Dado que los trabajos en clases pueden ser en pareja, si no tiene una computadora portátil, DEBE trabajar con un compañero que posea una o comunicarse con el profesor o ayudante para encontrar formas alternativas. Si falta su computadora portátil / teléfono en un día de clase específico puede acercarse personalmente al instructor al final de la clase, entregando lo solicitado en forma manual. Sin embargo, esto NO será permitido en reiteradas ocasiones.

Mecanismo de evaluación:
La nota final del curso se calcula de la siguiente manera:

$$NF = 0.2 * (P_1 + P_2) + 0.25 * P_3 + 0.1 * T + 0.1 * P + 0.15 * C$$

Donde:
- P_1 es la nota de la prueba 1
- P_2 es la nota de la prueba 2
- P_3 es la nota de la prueba 3 (en la fecha del examen)
- T es el promedio de las tareas
- P es el proyecto
- C es el promedio de los controles

Examen de repechaje
Si la Nota Final (NF) **es mayor o igual a 3.5 y menor a 4.0**, el alumno tendrá derecho a rendir un único examen de repechaje. **Si la nota obtenida es igual o superior a 4.0, entonces el alumno aprueba el curso con** $NF = 4.0$; independiente de la nota obtenida en el examen de repechaje. De lo contrario, la nota final del curso será la nota obtenida en el examen de repechaje.

Asistencia
Entendemos que todos podemos tener contratiempos que nos impidan ir a todas las clases. Sin embargo, al ser Programación un ramo que depende única y exclusivamente de la cantidad de horas dedicadas a la práctica y estudio, se les recomienda asistir a un 75% de las clases de cátedra.

En caso de inasistencia injustificada a una evaluación, el alumno recibirá la nota mínima (1.0). Si la inasistencia es debidamente justificada en Pregrado, la nota del examen reemplazará la nota de la evaluación no rendida.

Notas adicionales sobre el curso

1. El sitio Webcursos 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 el sitio 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. Use su teléfono celular de forma respetuosa durante la clase y solo cuando sea necesario.
    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. No solo distraes a tus compañeros, sino que puedes perderte algo importante :)
4. 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)

>> Material Web: La 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	Introducción a programación. Reglamentos y fechas de evaluación. Presentación de un video de ejemplo. Preguntar y analizar en forma breve cuál es la problemática. Introducción a algoritmos. Un caso de un problema que se resuelve con una secuencia de pasos, condiciones y ciclos que sean evidentes (estructurado). Problema estructurado vs no-estructurado, condiciones de borde.	Analizar y detectar la principal problemática de una película. Describir en un papel los pasos de cada una de las etapas del problema y su resolución. Discusión del caso presentado en clase. Preguntas posibles son ¿Qué es una instrucción? ¿Qué tanto detallada debe ser una instrucción? ¿Cómo reconocer condiciones? ¿Qué son los ciclos?	Pizarra Youtube Material de caso de estudio (video, texto) Cuaderno de apuntes
Semana 2	Definición formal de la resolución de problemas. Análisis en la formalización de un problema. ¿Cómo se describe un problema estructurado?, ¿cuáles son las condiciones de borde?, ¿estados y variables qué son? ¿Qué implicancias tiene el orden de la secuencia de pasos? ¿Si altero el orden, cambia el resultado?	Se describen uno o más ejemplos donde se describe un problema (ej. en un párrafo). Se pide sintetizar y detectar las condiciones antes descritas. Formalizar el problema.	Presentación Pizarra Cuaderno de apuntes
Semana 3	Descripción formal de un algoritmo mediante pseudo-código: pasar de un nivel genérico a un nivel lógico de programación. Ejemplos de algoritmos pasados a lenguajes de programación.	Actividades de descripción de un problema desde alto nivel hasta un nivel de detalle que permita programarlo. Un ejemplo son las recetas de cocina, formalismo matemáticos, etc.	Presentación Pizarra Cuaderno de apuntes
Semana 4	Introducción a Python. Tipos de herramientas y ambientes de trabajo. Manejo de IDE. Introducción de uso de variables, operadores lógicos y aritméticos.	Laboratorio de instalación y uso de IDE, donde los alumnos trabajan en una guía simple de ejemplos en sus computadores empleando Python (**Guía en CASA: Resolución de Guía**)	Presentación Laptop IDE Guía de ejercicios Cuaderno de apuntes
Semana 5	Uso y definición de condiciones en Python. Condicionales y ciclos simples. Solución de problemas simples con condicionales y ciclos en Python	En grupos de tres alumnos se resuelve un problema, y se presenta un código en clases. Guía de ejercicios guiado en clases para resolver dudas iniciales	Presentación Laptop IDE Cuaderno de apuntes
Semana 6	Uso de funciones pre-definidas en Python: print, input. Interacción con usuario. Debugging simple.	Usando Python interactuamos con el usuario mediante print e input. Creación de un bot simple usando Python. Luego usaremos print para hacer un debugging básico.	Presentación Laptop IDE Cuaderno de apuntes
Semana 7	Estructuras de datos en Python: listas, tuplas, sets y diccionarios. Almacenando y recuperando datos en las estructuras usando ciclos.	Usando Python y el módulo requests bajaremos páginas web, las guardaremos en archivos y las abriremos en el browser de los alumnos. Se generará una discusión ¿Por qué no se muestran las imágenes? Explicaremos como funciona un browser.	Presentación Laptop IDE Guía de ejercicios Cuaderno de apuntes
Semana 8	Manejo complejo con ciclos dobles en Python y condiciones anidadas. Solución de problemas orientados a la resolución de problemas con ciclos en Python	Alumnos trabajan en una guía simple de ejemplos en sus computadores empleando Python	Presentación Laptop IDE Cuaderno de apuntes
Semana 9	Manejo de Módulos: Llamado de funciones (random, math). Solución de problemas empleando funciones ya implementadas Python	En grupos de tres alumnos se resuelve un problema y se presenta un código en clases.	Pizarra Laptop IDE Cuaderno de apuntes

Semana 10	Introducción al análisis de datos con Numpy y Pandas: Manejo de vectores (1D, 2D existen) y operaciones simples	Se contrasta en clases la forma de calcular valores como el promedia de una lista mediante la forma tradicional vs Pandas. Discusión ¿cuáles son las ventajas, desventajas? ¿Debemos usar Pandas siempre? Guía de ejercicios	Presentación Laptop IDE Cuaderno de apuntes
Semana 11	Diseño e implementación de funciones y en Python.	Alumnos trabajan en una guía simple de ejemplos en sus computadores empleando Python	Presentación Laptop IDE Cuaderno de apuntes
Semana 12	Desarrollo de aplicaciones en Python. Trabajo guiado en clase.	El temario es dependiente de la elección de proyecto.	Presentación Laptop IDE Cuaderno de apuntes
Semana 13	Desarrollo de aplicaciones en Python. Trabajo guiado en clase.	El temario es dependiente de la elección de proyecto.	Presentación Laptop IDE Cuaderno de apuntes
Semana 14	Desarrollo de aplicaciones en Python. Trabajo guiado en clase.	El temario es dependiente de la elección de proyecto.	Presentación Laptop IDE Cuaderno de apuntes
Semana 15	Presentación de proyectos	Alumnos demuestran sus proyectos en la comunidad	Presentación Laptop

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 para cada objetivo descrito

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 estudiante debería demostrar para aprobar el curso. Indicar procedimiento de evaluación: trabajo, prueba, interrogación, etc.	**Clases** Indicar número de clase o semana o fecha donde se trata el Objetivo de acuerdo a la planificación
(a) Aplicar conocimiento de matemáticas, ciencia e ingeniería	Modelar la resolución de problemas estructurados a través del diseño e implementación de un lenguaje de programación.			Trabajo personal y grupal; Prueba.	
(b) Diseñar y conducir experimentos, como también para analizar e interpretar datos					
(c) Diseñar un sistema, componente o proceso que cumpla con las necesidades requeridas, considerando restricciones realistas	Diseñar, proponer e interpretar códigos de aplicaciones computacionales escritas en un lenguaje de programación determinado.			Trabajo personal y grupal; Prueba.	
(d) Trabajar en equipos multidisciplinarios					
(e) Identificar, formular y resolver problemas de ingeniería	Modelar la solución a problemas simples descritos en lenguaje natural mediante la construcción de un diagrama de flujo o de actividad que represente la secuencia de pasos que se deben ejecutar para resolver el problema.			Trabajo personal y grupal; Prueba.	
(f) Comprender la responsabilidad ética profesional					

(g) Comunicar efectivamente					
(h) Comprender el impacto de las soluciones de ingeniería en un contexto global, económico, ambiental y social					
(i) Reconocer la necesidad y la habilidad para comprometerse en un aprendizaje permanente					
(j) Conocer temas contemporáneos					
(k) Usar técnicas, habilidades y herramientas modernas de ingeniería necesarias para la práctica ingenieril					