UCR ECCI Escuela de Ciencias de la Computación e Informática

CI-0202
Principios de informática
I ciclo 2019

Grupo 01: [M 13:00 – 16:50/ Lab. Industrial]
Profesor: [M.c.i. Alejandro Ulate Campos]
Correo: [alejandro.ulate@ucrso.info]
Consulta: [S 13:00 – 16:50/Virtual]

Carta al estudiante. Curso de servicio, sin requisitos ni correquisitos. 4 créditos, 4 horas presenciales, 8 horas extraclase

Descripción del curso

Es un curso básico de programación para estudiantes del área de ingeniería y afines. En el curso se introduce al estudiante al pensamiento abstracto para la resolución de problemas de ingeniería y científicos, automatizable por medio de herramientas informáticas de desarrollo, utilizando metodologías sistemáticas. El estudiante aprenderá a reconocer la aplicabilidad de flujos de control y modelos de datos básicos para lograr el diseño e implementación de programas y algoritmos.

Objetivos

Proveer formación básica en programación y construcción de algoritmos y de programas, para la resolución de problemas utilizando técnicas actuales. Al finalizar este curso el estudiante será capaz de:

1. Diseñar, organizar e implementar algoritmos para resolver problemas específicos del área de ingeniería, ciencias y afines.
2. Usar un ambiente de programación para la edición, prueba y depuración de programas.
3. Reutilizar componentes de software.
4. Aplicar buenas prácticas de construcción de software.

Contenidos y cronograma

	Semana
1. Fundamentos de la programación	**11-mar**
Lenguajes de programación: concepto de programación, lenguaje máquina, lenguaje ensamblador, lenguaje de alto nivel, maquina virtual, compilador y paradigmas. Ciclo de vida de un programa: problema, análisis, diseño, implementación y prueba Algoritmo: concepto, primitivas y ejemplos	
2. Introducción a la programación orientada a objetos	**18-mar**
Paradigma: clases e instancias, atributos y métodos, abstracción y reutilización Análisis y diseño: modelaje de clases e instancias Construcción (compilación, interpretación) y ejecución	
3. Sistemas numéricos y representación de datos	**25-mar**
Bases y conversión: decimal, binaria y hexadecimal Sistemas de codificación de texto: ASCII y UNICODE	
4. Tipos de datos	**25-mar**
Tipos de datos: enteros, reales, booleanos, caracteres e hileras de texto. Aritmética de precisión entera, aritmética de precisión flotante.	
5. Definición y utilización de variables	**25-mar**
Definición de valores y variables: Declaración: tipo, identificador y dirección Inicialización de acuerdo al tipo de datos: valores, indirección (ej: punteros, referencias) Estado de la variable en memoria (valor) Asignación y conversión de tipos Utilización de variables: Atributos de clase: declaración, ámbito de vida y ocultamiento (encapsulamiento) Variables locales: declaración y ámbito de vida Estáticas y constantes: declaración y ámbito de vida	
6. Entrada y salida básica, verificación	**01-abr**
Entrada y salida básica: Entrada: parámetros de línea de comandos y entrada interactiva Salida: salida estándar y salida interactiva Verificación de datos y manejo de errores (ej: manejo de excepciones)	

7. Expresiones y operadores	**01-abr**
Evaluación de expresiones y orden de precedencia Aridad de operadores: unarios (ej: negación) y binarios (ej: multiplicativos y aditivos) Tipos de operadores: aritméticos, relacionales (comparación e igualdad), lógicos, y de asignación	
8. Instrucciones y estructuras de control	**08, 22-abr**
Secuencia de instrucciones y bloques de instrucciones Condición de instrucciones Repetición de instrucciones por condición y por contador	(15: sem. santa) (22: sem. unives.)
9.a. Subrutinas: fundamentos	**29-abr a 06-may**
Conceptos: modularización y reutilización, declaración e invocación Componentes: encabezado (identificador, parámetros y tipo de retorno) y cuerpo Funciones y procedimientos Sobrecarga de subrutinas: declaración y resolución de llamados	
9.b. Subrutinas: funcionamiento	**13-may**
Paso de argumentos: por valor y por indirección Alojamiento estático de memoria, alojamiento dinámico de memoria, y alojamiento automático de memoria (pila de llamados) Reglas de alcance o ámbito de identificadores Constructores: concepto y utilización, declaración e invocación	I examen (hasta tema 8)
10. Recursividad	**13-may**
Concepto y utilización Orden de llamados	
11a. Colecciones lineales de datos: fundamentos	**20-may**
Concepto (ej: arreglos, listas), estructura y estado de memoria Declaración e inicialización Acceso a elementos y recorrido Paso de colecciones por argumento a subrutinas	
11b. Colecciones lineales de datos: operaciones	**27-may**
Utilidad y operaciones comunes (suma, promedio, mínimo, máximo) Búsqueda (de acuerdo al tipo de datos de los elementos) Ordenamiento (de acuerdo al tipo de datos de los elementos)	II examen (hasta tema 10)
12. Matrices	**03-jun**
Concepto, estructura y estado de memoria Declaración e inicialización Acceso a elementos y recorrido	
13. Hileras o cadenas de caracteres (textos)	**10-jun**
Concepto Operaciones: concatenación, obtener tamaño, extraer carácter o fragmento, comparación, búsqueda, reemplazo, conversión a mayúscula o minúscula, conversión a arreglo	
14. Entrada y salida: archivos	**17-jun**
Conceptos y organización física de archivos Operaciones de archivos: lectura y escritura Procesamiento binario/textual: apertura/cierre y lectura/escritura	
15. Programación avanzada (tema a escoger entre los siguientes)	**24-jun a 01-jul**
Matrices, algoritmos y bibliotecas de álgebra lineal. Fundamentos de graficación y de interfaces gráficas. Indirección (punteros y referencias), copia y clonación de objetos. Algoritmos de búsqueda y ordenamiento básicos. Herencia y polimorfismo Aritmética de precisión arbitraria. Cuadernos de lenguajes de computación. Visualización de datos (ej: creación de gráficas, animaciones). Simulación (ej: a partir de modelos matemáticos, físicos). Predicción (usando modelos simples).	III examen (hasta tema 14)

Metodología y evaluación [Combinación magistral y aprender/haciendo]

Se sigue una metodología híbrida tradicional-constructivista. En cada semana el estudiante dedica 12 horas al curso de Principios de informática, 4 presenciales y 8 extra-clase. Las 4 horas presenciales a la semana se dividen en 2 horas de lecciones magistrales y 2 en el laboratorio de cómputo. En las lecciones magistrales se utilizan recursos audiovisuales y la pizarra para ilustrar el proceso de resolución de problemas y los conceptos de programación. En las horas de laboratorio los estudiantes resuelven problemas cortos de programación que requieren la teoría impartida en las clases magistrales con el apoyo del profesor del curso y el asistente asignado.

Actividad	Peso
3 exámenes parciales	60%
2 Laboratorios 10%c/u	20%
Investigación	20%

En las 8 lecciones extra-clase a la semana el estudiante resuelve problemas de programación planteados en forma de tareas cortas individuales de igual ponderación. Los enunciados de las tareas serán provistos por el profesor en un aula virtual alojada en Mediación Virtual (http://mediacionvirtual.ucr.ac.cr/). El estudiante entregará sus soluciones en el aula virtual antes de la fecha límite estipulada en el enunciado de cada tarea. En caso de una entrega de h horas tardías se aplicará un castigo de $h^{3/2}$ puntos. El profesor proveerá en las primeras lecciones del curso indicaciones para acceder al aula virtual.

Se realizarán tres exámenes parciales de 20% cada uno en las semanas indicadas en el cronograma a la derecha de los contenidos. Los exámenes se realizarán en computadora durante la lección en el laboratorio y por tanto tendrán una duración máxima de 1 hora con 40 minutos. En caso de desfase, el profesor ajustará las fechas del cronograma y negociará las de los exámenes con los estudiantes una vez que se hayan cubierto los contenidos del mismo. Por la naturaleza de los contenidos, los exámenes son acumulativos. Cada examen planteará un problema que el estudiante debe resolver mediante la programación de computadoras. El profesor realizará una práctica para el primer examen parcial con el fin de que los estudiantes puedan familiarizarse con este tipo de evaluación.

El profesor podrá realizar exámenes cortos (quices) en cualquiera de las clases magistrales. Además, se realizarán trabajos en clase que se evaluarán durante la permanencia en el laboratorio. De acuerdo al artículo 15 del Reglamento de Régimen Académico Estudiantil dichas reglas quedan establecidas después de este aviso.

Lineamientos

1. Durante los exámenes el estudiante podrá consultar material teórico (libros, apuntes) o código que sea producto de su trabajo en el curso o que haya sido proporcionado por el profesor. Los medios para consultar estos materiales durante el examen serán indicados previamente por el profesor.
2. Es ilegal presentar como propio, código parcial o total escrito por otras personas u obtenido de fuentes de información, como por ejemplo de libros o de Internet, sin la autorización expresa del docente. Los exámenes son estrictamente individuales, y en caso de ser por computadora, es ilegal violar esta regla a través de aplicaciones de correo electrónico o mensajería. En cualquier asignación en que se detecte plagio, se asignará un cero como nota en la evaluación y se aplicará la normativa estipulada en el Reglamento de Orden y Disciplina de los Estudiantes de la Universidad de Costa Rica (http://www.cu.ucr.ac.cr/uploads/tx_ucruniversitycouncildatabases/normative/orden_y_disciplina.pdf).
3. Todo código entregado por el estudiante debe compilar sin errores. De lo contrario será calificado con 0.
4. En cualquiera de los tipos de evaluaciones, el profesor podría proponer actividades opcionales por crédito extra en la nota del curso.

Bibliografía

1. Rossant, Cyrille, “Learning IPython for Interactive Computing and Data Visualization”, ISBN: 978-1782169949, PACKT Books, 2013. https://www.packtpub.com/big-data-and-business-intelligence/learning-ipython-interactive-computing-and-data-visualization
2. Wachenchauzer, Rosita, Manterola, Margarita y otros, “Algoritmos y Programación con lenguaje Python”, OpenLibra, 2011. https://openlibra.com/es/book/algoritmos-y-programacion-con-lenguaje-python

Software

1. **Python** versión 3, Descargas: https://www.python.org/downloads/
2. VSCode Editor/IDE para Python (https://code.visualstudio.com/download)
3. Cuaderno **Jupyter** es una aplicación web que permite crear y compartir documentos que contienen código en vivo, ecuaciones, visualizaciones y texto explicativo. Sus usos incluyen: limpieza y transformación de datos, simulaciones numéricas y mucho más: http://jupyter.org/
4. **Opcional - Anaconda** (incluye Python y Jupyter): https://www.continuum.io/downloads