UCR ECCI Escuela de Ciencias de la Computación e Informática

CI-0202
Principios de informática
I ciclo 2018

Grupo 04: L9-11 Sala1-IN / J9-11 103-IF
Profesor: Jeisson Hidalgo-Céspedes
Correo: jeisson.hidalgo@ucr.ac.cr
Consulta: KV 8-10 201-IF

Carta al estudiante. Curso de servicio, sin requisitos ni correquisitos. 4 créditos, 4 horas presenciales, 8 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	**12-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	**11 a 18-jun**
Paradigma: clases e instancias, atributos y métodos, abstracción y reutilización Análisis y diseño: modelaje de clases e instancias Compilación y ejecución	
3. Sistemas numéricos y representación de datos	**02-abr**
Bases y conversión: decimal, binaria y hexadecimal Sistemas de codificación: ASCII y UNICODE	
4. Tipos de datos	**19-mar**
Tipos de datos: primitivos (enteros, reales, booleano y caracteres) e hileras Precisión: entero (byte, short, int y long), real (float y double), booleano, carácter e hilera (ejemplo: secuencias de escape)	
5. Definición y utilización de variables	**19-mar**
Definición de variables: Declaración: tipo, identificador y dirección Inicialización: tipo primitivo (valor), instancia (referencia) y estado de memoria Asignación y conversión (i.e. type casting) 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, manejo de excepciones	**19-mar**
Entrada y salida básica: Entrada: parámetros de línea de comandos y diálogo Salida: línea de comandos y diálogo	

Manejo de excepciones: Concepto, ejemplos y definición (clase), lanzamiento y atrape	
7. Expresiones y operadores	**02-abr**
Aritméticos binarios (multiplicativos y aditivos) y unarios (negación y posfijos) Relacionales (comparación e igualdad), lógicos (binarios y unarios) y asignación Evaluación y orden de precedencia	
8. Instrucciones y estructuras de control	**09 a 16-abr**
Estructuras de secuenciación ({}) Estructuras de selección o bifurcación (if/else y switch) Estructuras de repetición o iteración (while, do y for)	
9.a. Métodos: Fundamentos	**30-abr**
Conceptos: modularización y reutilización, declaración e invocación Componentes: encabezado (identificador, parámetros y tipo de retorno) y cuerpo Métodos estáticos (funciones) Sobrecarga: declaración, firmas y resolución de llamados	
9.b. Métodos: Funcionamiento	**07-may**
Paso de argumentos: por valor y por referencia Estado de memoria estática, memoria dinámica y pila de llamados Reglas de alcance o ámbito de identificadores Constructores: Concepto y utilización, declaración e invocación	I examen (hasta tema 8)
10. Recursividad	**14-may**
Concepto y utilización Orden de llamados	
11a. Arreglos o vectores: fundamentos	**21-may**
Concepto, estructura y estado de memoria Declaración e inicialización Acceso a celdas y recorrido Parámetros de tipo arreglo y paso de argumentos	
11b. Arreglos o vectores: operaciones	**28-may**
Utilidad y operaciones comunes (suma, promedio, mínimo, máximo) Búsqueda: primitivos e instancias Ordenamiento: primitivos e instancias	II examen (hasta tema 10)
12. Matrices	**04-jun**
Concepto, estructura y estado de memoria Declaración e inicialización Acceso a celdas y recorrido	
13. Hileras o cadenas de caracteres (textos)	**11-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	**18-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)	**25-jun a 02-jul**
Matrices, algoritmos y bibliotecas de álgebra lineal. Fundamentos de graficación y de interfaces gráficas. Punteros y referencias, copia y clonación de objetos. Algoritmos de búsqueda y ordenamiento básicos. Herencia y polimorfismo Cuadernos de lenguajes de computación.	III examen (hasta tema 14)

Metodología y evaluación

En cada semana se imparten cuatro horas, dos magistrales y otras dos en laboratorio de cómputo. En las lecciones magistrales se utilizan recursos audiovisuales y la pizarra para ilustrar conceptos e implementaciones. En las lecciones de laboratorio los estudiantes resuelven e implementan ejercicios cortos de programación relacionados con la teoría impartida en las clases magistrales.

Actividad	Peso
3 exámenes parciales	60%
Ejercicios	40%

La logística para realizar los exámenes parciales, por ejemplo en papel o con herramientas digitales, es escogido por el profesor y comunicado a los estudiantes durante las primeras lecciones del curso.

En las 8 horas extraclase, los estudiantes resolverán ejercicios propuestos en una plataforma de juez automático. Los ejercicios tendrán el formato usado en los concursos de programación de la ACM. Se utilizará como juez automático la plataforma HackerRank [https://www.hackerrank.com]. Los estudiantes proveerán su correo electrónico al profesor y recibirán invitaciones por este medio para acceder a los ejercicios.

Los ejercicios están organizados en secciones temáticas. Aproximadamente cada dos semanas del curso se habilitarán los ejercicios de una sección y se enviarán invitaciones a los estudiantes. Durante las lecciones de la semana correspondiente, el profesor explicará, al menos parcialmente, los conceptos de programación requeridos por los ejercicios. Algunos ejercicios requerirán más detalles o conceptos de los vistos en clase, con el fin de que el estudiante investigue otras fuentes de información para poder resolver el problema y desarrollar habilidades autodidácticas requeridas en el ejercicio de la profesión.

Los estudiantes resolverán los ejercicios en horas extraclase y durante las lecciones presenciales de laboratorio. Cada vez que el estudiante resuelve un ejercicio de programación obtendrá puntos que irá acumulando en la plataforma HackerRank. El estudiante recibirá crédito por sus puntos de HackerRank hasta un máximo de 40% de la nota del curso.

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

Java:

1. Barnes, David J. & Kölling, Michael, “Programación orientada a objetos con Java”, ISBN: 978-84-8322-350-5, Pearson Educación, 2007. http://www.bluej.org/objects-first/
2. Ceballos, Francisco Javier, “Java 2 - Curso de Programación - 3º ed.”, ISBN 970-15-1164-6, Alfaomega Ra-Ma, 2006. http://fjceballos.es/
3. Deitel, Paul; Deitel, Harvey. “Cómo programar en Java, décima edición”, Pearson Educación, México, 2016.
4. David J. Eck, “Introduction to Programming Using Java”, Seven Edition. Libro libre disponible en http://math.hws.edu/javanotes/.

Software

Java:

1. **Java SE** Descargas: http://www.oracle.com/technetwork/java/javase/downloads/
2. **Jeliot 3** es una aplicación para la Visualización del Programa: http://cs.joensuu.fi/jeliot/
3. **DrJava** es un ambiente de desarrollo liviano para la escritura de programas Java: http://www.drjava.org
4. **BlueJ** es un ambiente Java integrados diseñado específicamente para la enseñanza introductoria: http://www.bluej.org/
5. **Greenfoot** es un ambiente de desarrollo orientado a aplicaciones gráficas sencillas: http://www.greenfoot.org/
6. **JAMA**, es un paquete de Matrices Java: http://math.nist.gov/javanumerics/jama/
7. **Chart2D** es una biblioteca funcional minimalística para graficación: http://jchart2d.sourceforge.net/
8. **Netbeans** es un ambiente de desarrollo avanzado: http://www.netbeans.org/.
9. **Eclipse** es un ambiente de desarrollo avanzado: http://www.eclipse.org/ide/.