IE-0217 Estructuras de Datos Abstractas y Algoritmos para Ingeniería

Carta al estudiante

Curso Semestral Virtual

Nombre: Estructuras de Datos Abstractas y Algoritmos para Ingeniería

Sigla: IE-0217

Ciclo: 2 – 2022

Instructor: Richard Marín Benavides

Coordinador: José Freddy Rojas Chavarría

Modalidad: Virtual

Metodología: Este curso es virtual para su versión II semestre 2022. Se utilizará la plataforma institucional Mediación Virtual para colocar los documentos, presentaciones y vídeos del curso. Las clases asincrónicas se realizarán por medio de tareas, exámenes y foros, y las actividades sincrónicas por medio de la aplicación Zoom. Para las clases sincrónicas se compartirá con antelación la fecha y el enlace a utilizar, que permita al estudiante preparar su espacio físico y dispositivos necesarios, se utilizará la herramienta GIT y un repositorio para la administración del material del curso y la entrega de trabajos.

Horas: 6 horas semanales los días Miércoles y Viernes de 6:00 a 9:00 p.m

Número de créditos: 3

Requisitos: IE-0117

Clasificación: Propio

Ingreso Mediación Virtual: IE0217_02

Descripción:

El curso de Estructuras Abstractas de Datos y Algoritmos para Ingeniería tiene la finalidad de ahondar en conceptos de programación, lo que les permite a los estudiantes enfrentarse sistemáticamente al desarrollo de soluciones computacionales a problemas, mediante la selección del lenguaje de programación y las estructuras de datos y algoritmos apropiados en cada situación. Al finalizar el curso, los estudiantes comprenderán los fundamentos teóricos y prácticos del uso de los lenguajes Python y C++, además del uso de herramientas de control de versiones. Podrán, además,

comprender las estructuras de datos y algoritmos más utilizados. Además, estarán en capacidad de proponer nuevas estructuras de datos y algoritmos para la resolución de problemas específicos.

Objetivo General.

Alcanzar el conocimiento suficiente para la implementación de algoritmos y selección de estructuras de datos para poder resolver computacionalmente problemas de ingeniería, utilizando lenguajes acordes a la problemática a resolver y poder trabajar con control de versiones en equipos de trabajo de desarrollo de software para ingeniería.

Objetivos Específicos

- Desarrollar habilidades de programación en lenguaje C/C++
- Desarrollar habilidades de programación en lenguaje Python.
- Dominar y desarrollar soluciones aplicadas con programación orientada a objetos.
- Utilizar herramientas para el manejo de estructuras dinámicas de datos como STL para C++ y listas y diccionarios en Python.
- Conocer los conceptos de las estructuras de datos: arreglos, vectores, listas, colas, pilas, hash, conjuntos(set), mapas y árboles.
- Manejar contenedores de datos y el uso de iteradores.
- Desarrollar algoritmos de búsqueda y ordenamiento.
- Conocer criterios básicos de compiladores e interpretadores para usos asociados a protocolos y operaciones complejas.
- Desarrollar la habilidad de controlar versiones de desarrollo de software mediante el uso de la herramienta GIT y la aplicación de interfaces visuales para el manejo de reservorios locales o globales como GitHub.
- Desarrollar habilidades de depuración básica de programas.

Ejes Temáticos y contenidos

Módulo 1. Herramientas y contextualización de espacio de desarrollo.

1. Herramientas y entorno del sistema operativo.
2. Instalación de Code::Blocks y MS Visual Studio Code para C/C++ y Python
3. Instalación y uso de herramienta GIT y uso de comandos de línea.
    a. GIT en MS Windows.
    b. GIT en Linux.
    c. GIT en OSX.
    d. Herramienta comercial alternativa SmartGitHG y su uso en estudiantes.
4. Entorno de desarrollo para ingeniería de software.
5. Herramientas de visualización e intercambio: UML y XML.
6. Estándares y modelos internacionales.

Módulo 2: Control de versiones

1. Criterios de control de versiones.
2. GIT
    a. Conceptos de uso
    b. Reservorio local
    c. Reservorio remoto
        i. GitHub
3. Conceptos GIT (inglés)
    a. Create
    b. Clone
    c. Working directory
    d. Stage área
    e. Local repo
    f. Remote repo
    g. Commands
        i. add
        ii. commit
        iii. push
        iv. pull
        v. checkout
4. Herramienta comercial (uso libre para estudiantes)
    a. SmarGitHG
        i. Uso restringido
        ii. Criterios de uso

Módulo 3. Lenguaje para programación C/C++

1. Introducción al lenguaje de programación: historia, filosofía, interprete, entre otros.
2. Herramientas de desarrollo. Flujo de desarrollo (compilación, interpretación, etc.).
3. Características básicas del lenguaje: tipos de datos, estructuras de control, etc.
4. Aplicación de la programación orientada a objetos al lenguaje
5. Técnicas particulares de programación para el lenguaje a. C++:
    a. Entorno de consola.
    b. Entorno IDE.
    c. Compilación
        i. Objetos.
        ii. Librerías.
        iii. Ejecutables.
        iv. Librerías dinámicas.
    d. Manejo de memoria
    e. Errores típicos de compilación
    f. Automatización del proceso de compilación (Makefiles, CMake, Autotools)
    g. La librería estándar: STL, Boost.

Módulo 4: OOP programación orientada a objetos en C/C++

1. Concepto de objeto en programación.
2. Declaración de clase
3. Instancia de objetos.
4. Atributos y métodos.
5. Jerarquía y herencia.
6. Polimorfismo.
7. Modelo de diseño orientado a objetos.

Módulo 5: Estructuras dinámicas de datos y criterios de algoritmos.

1. Complejidad computacional y eficiencia de algoritmos.
2. Corrección de algoritmos.
3. Tipos de datos abstractos.
4. Estructuras de datos:
5. Importancia de las estructuras de datos
    a. Memory leak
    b. Arreglo
    c. Vector
    d. Lista
    e. Pila
    f. Colas
    g. Secuencias (Hash)
    h. Conjuntos
    i. Aboles y criterios de diseño y uso
    j. Grafos
6. Búsquedas.
7. Ordenamiento.
8. Algoritmos: criterios de diseño de un algoritmo.
    a. Métricas de diseño.
    b. Eficiencia de un algoritmo
9. Bibliotecas: STL, Boost.

Módulo 6: Lenguaje de Programación Python.

1. Paradigma de Python y diferencias con C/C++
    a. Python 2.7
    b. Python 3.x
2. Interprete de Python.
3. Uso de MS Visual Code
4. Criterios de uso C++ vs Python.
    a. Uso de Python como herramienta de intercambio.
5. Otros criterios: Lenguaje de Minería de datos R y diferencias con Python.
6. Herramientas de desarrollo.
    a. Anaconda

    b. Spyder
    c. Eclipse
7. Flujo de desarrollo
8. Características básicas del lenguaje: tipos de datos, estructuras de control, etc.
9. Aplicación de la programación orientada a objetos al lenguaje en Python.
10. OOP en Python, ventajas y desventajas
11. Estructuras de datos propias de Python.

Módulo 7: Criterios básicos de minería de datos y uso de Python en el entorno mundial

1. Criterios de ciencia de datos y minería de datos.
2. Conceptos de clasificación (clustering)
    a. Usos de Python para manejo de datos masivos.
    b. Librerías de Python para manejo matemático.
    c. Librería Pandas y estructuras de datos.
3. Conceptos de predicción de series de tiempo
    a. Criterios generales

Asuntos metodológicos

Este curso es virtual en su versión del II semestre 2022. Se usarán tres canales de comunicación: el sitio de mediación virtual de la UCR, la plataforma ZOOM para la impartición de clases y consultas y un repositorio GIT para la entrega de tareas, exámenes y los resultados de los mismos.

La plataforma institucional Mediación Virtual será el canal de comunicaciones y además se colocarán los documentos, presentaciones y vídeos del curso.

Las consultas se deben hacer por el sitio de mediación virtual.

Para la impartición de clases se habilitará una sesión en Zoom que será aplicable a lo largo del semestre, el link de esta sesión se comunicará por Mediación Virtual. También se utilizará un link adicional para las consultas del examen final, este será enviado la semana anterior a dicho examen.

Las clases serán sincrónicas y asincrónicas y en el caso de las sesiones asincrónicas, estas se realizarán por medio de tareas, exámenes y foros. Las actividades sincrónicas por medio de la aplicación “Zoom”.

Para las clases sincrónicas se compartirá con antelación la fecha y el enlace a utilizar, que permita al estudiante preparar su espacio físico y dispositivos necesarios. Además, se desarrollarán los criterios con el apoyo de una pizarra virtual, misma que estará disponible para descargar una vez concluida la clase.

Se usará un reservorio GIT para el acceso a los archivos de código fuente y la entrega de trabajos y tareas. Este será debidamente detallado en la primera clase. El curso consta de un conjunto completo de carpetas con *Código Fuente* para cada clase y que abarca el tema desarrollado tanto en C/C++ como en Python. Todos los accesos a este código, así como la entrega y calificación de tareas se realizará con el repositorio GIT.

El enfoque del curso será orientado a la resolución de problemas a partir de los criterios mostrados en clase y se asignarán tareas individuales o en parejas (dependiendo del tema). Las clases se impartirán en forma sincronizada con presencia del profesor y los estudiantes en la mayoría de las lecciones. En los casos que amerite se asignarán clases asincrónicas, en las que se dejará el material accesible a los estudiantes y con este deben desarrollar la solución de un problema que se revisará la siguiente clase.

También en el curso se desarrollará un proyecto final que constituye un caso de uso que se apoyará en las clases del curso. El proyecto se presentará en la última semana de clases y es requisito presentarlo para hacer el examen final.

Los estudiantes deberán plantear propuestas de proyectos a desarrollar que resuelvan problemas reales utilizando el conocimiento aprendido durante el curso, para lo que se solicitarán proyectos de investigación, diseño e implementación, con la respectiva presentación oral y escrita de sus resultados.

Todas las tareas y trabajos se manejarán por la plataforma del repositorio GIT que será brindado por el profesor.

Al ser en modalidad virtual cada estudiante deberá contar con su computador propio para el desarrollo del curso. El estudiante podrá trabajar en Windows 10 o en Linux (por ejemplo, Debian 10)

Evaluación

- Proyecto 30%
- Tareas programadas 40%
- Examen Final 30%

Bibliografía

1. Narasimha Karumanchi, Data Structures and Algorithms Made Easy (C/C++), 5th ed.,Career Monk, 2017.
2. Mark A. Weiss, Data Structures & Algorithms Analysis in C++, 4th ed. Addison-Wesley Longman, Inc., 2014.
3. Mark A. Weiss, Data Structures & Algorithms Solving Using C++, 2nd ed. Addison-Wesley Longman, Inc., 2000.
4. Alfred V. Aho, John E. Hopcroft, and Jeffrey Ullman. Data Structures and Algorithms.

Addison-Wesley Longman Publishing Co., Inc., 1983.
5. T. Cormen. Introduction to Algorithms. The MIT Press, 2001.
6. William H. Ford. Data structures with C++ using STL. Prentice Hall, 2001.
7. Michael Goodrich. Algorithm Design. John Wiley and sons, 2002.
8. Nicolai M. Jossutis. The C++ standard library, a tutorial and reference. Addison Wesley, 1999.
9. Luis Joyanes. Estructuras de datos en C++. McGraw Hill, 2007.
10. Yedidyah Langsam and Moshe J. Augenstein. Estructuras de datos con C y C++. Prentice Hall
11. Kernighan, Brian y Pike, Rob. El entorno de programación de Unix. Prentice Hall, 1984. 12. Python Documentation. http://docs.python.org/.
13. González Duque, R. Python para todos. http://mundogeek.net/tutorial-python/.
14. Swaroop C. H. A Byte of Python. http://www.swaroopch.com/notes/Python.
15. Matloff N. A Quick, Painless Tutorial on the Python Language. http://heather.cs.ucdavis.edu/~matloff/python.html.
16. Downey A. Think Python: How to Think Like a Computer Scientist. http://www.greenteapress.com/thinkpython/thinkpython.html.
17. Severance C. Python for Informatics: Exploring Information. http://www.py4inf.com/book.php.
18. Shaw Z. A. Learn Python the Hard Way. http://learnpythonthehardway.org/.
19. Schildt, H. C/C++ Programmer’s Reference. Segunda edición. New York: McGraw-Hill