2234 – TECNOLOGÍA ORIENTADA A OBJETOS

UASLP
Universidad Autónoma de San Luis Potosí

FACULTAD DE INGENIERÍA
Área de Ciencias de la Computación

Clave de la materia: 2234
Clave Facultad: 2234
Clave U.A.S.L.P.: ---- **Clave CACEI:** IA
Nivel del Plan de Estudios: 4 **No. de créditos:** 8
Horas/Clase/Semana: 4 **Horas totales/Semestre:** 64
Horas/Práctica (y/o Laboratorio): 0
Prácticas complementarias: 0
Trabajo extra-clase Horas/Semana: 4
Carrera/Tipo de materia: I.S.I., I.C., I.I./Obligatoria
No. de créditos aprobados: ----
Fecha última de Revisión Curricular: 29/Noviembre/2018
Materia y clave de la materia requisito: 2232 – Estructuras de Datos II
2304 – Ingeniería de Software

OBJETIVO DEL CURSO

Comprender el paradigma orientado a objetos, y aplicar sus tecnologías, mecanismos, métodos de diseño y programación en el desarrollo de aplicaciones de software.

CONTENIDO TEMÁTICO

1. INTRODUCCIÓN

Tiempo Estimado: 2 hrs.

Objetivo: Conocer los conceptos fundamentales del paradigma de programación orientada a objetos, sus mecanismos y lenguajes que le dan soporte, además de sus ventajas y aplicaciones.

1.1. Paradigma de programación orientado a objetos
1.2. Lenguajes de programación orientados a objetos
1.3. Ventajas y aplicaciones de la programación orientada a objetos

2. INTERACCIÓN DE OBJETOS

Tiempo Estimado: 4 hrs.

Objetivo: Comprender las características y las diferencias que existen entre los objetos y las clases, así como también entre los mensajes y los métodos.

2.1. Creación y uso de objetos
2.2. Envío de mensajes e invocación de métodos
2.3. Constructores y métodos de acceso/modificación
2.4. Parámetros, variables temporales y de instancia
2.5. Alcance y tiempo de vida de un objeto

3. ESTRUCTURA DE CLASES

Tiempo estimado: 10 hrs.

Objetivo: Comprender la estructura interna de una clase y aplicar la encapsulación para su implementación; así como conocer y utilizar colecciones e iteradores para resolver problemas que involucren un grupo de objetos.

3.1. Definición y estructura de las clases
3.2. Encapsulación, abstracción y modularización
3.3. Diagramas de clases y diagramas de objetos
3.4. Tipos nativos y referencias a objetos
3.5. Colecciones e iteradores

4. HERENCIA Y POLIMORFISMO

Tiempo estimado: 16 hrs.

Objetivo: Aplicar el mecanismo de herencia y polimorfismo para mejorar la estructura de un programa orientado a objetos; además de comprender el uso de las clases abstractas y las interfaces.

4.1. Subclases, herencia y sobreescritura de métodos
4.2. Polimorfismo y ejecución dinámica de métodos
4.3. Tipos, subtipos y conversiones
4.4. Clases abstractas
4.5. Interfaces

5. PRUEBAS Y MANEJO DE ERRORES

Tiempo estimado: 16 hrs.

Objetivo: Aplicar las técnicas más adecuadas para la prevención, detección y recuperación de errores.

5.1. Pruebas unitarias y automatizadas
5.2. Programación defensiva e informe de errores
5.3. Lanzamiento y manejo de excepciones
5.4. Excepciones comprobadas y no comprobadas

6. DISEÑO DE CLASES Y APLICACIONES

Tiempo estimado: 16 hrs.

Objetivo: Comprender los principios, patrones y prácticas de diseño requeridos para la construcción de aplicaciones orientadas a objetos.

6.1. Diseño dirigido por responsabilidades
6.2. Cohesión y acoplamiento
6.3. Refactorización
6.4. Principios S.O.L.I.D.

METODOLOGÍA

Explicación de los conceptos por parte del profesor utilizando medios didácticos y actividades de implementación en clase. Desarrollo de algoritmos y ejercicios por parte de los alumnos siguiendo el método de aprendizaje basado en problemas, utilizando diagramas de clase y escribiendo paso por paso el código de la implementación. Se espera que el alumno investigue ciertos temas, siguiendo el método de aula invertida. Implementar estrategias de trabajo en equipo cuando sea conveniente (aprendizaje colaborativo). Utilizar herramientas de control de versiones de código (git), y repositorios centralizados de código (GitHub) para llevar un registro y control de todos los ejemplos realizados en clase, ejercicios, tareas y proyectos.

EVALUACIÓN

Se realizarán cuatro exámenes parciales de forma colegiada en las fechas establecidas por la Facultad, de acuerdo al Reglamento de Exámenes. La calificación de los exámenes parciales estará compuesta por un examen, otras actividades (tareas, investigaciones, resolución de problemas, ejercicios, etc.), y la realización de un proyecto de programación. La entrega en tiempo y forma de dicho proyecto es requisito para tener derecho a cualquier calificación final. La calificación del examen ordinario es el promedio de los cuatro parciales.

BIBLIOGRAFÍA

Bibliografía Básica

Barnes, D. *Programación Orientada a Objetos con BlueJ*. 6ta Edición. Pearson Educación, 2017.

Hortsmann, Cay S. *Big Java Early Objects*. 6th Edition. Wiley, 2017.

Weisfeld, Matt. *The Object-Oriented Thought Process*. 4th Edition. Addison-Wesley Professional, 2013.

Joyanes Aguilar, L., I. Zahonero M. *Programación en C, C++, Java y UML*. McGraw-Hill, 2014.

Bibliografía Complementaria

Martin, Robert C. *Agile Software Development, Principles, Patterns, and Practices*. International Edition, Prentice Hall, 2013.

Dennis, A., Haley, B. *Systems Analysis and Design: An Object-Oriented Approach with UML*. 6th Edition, Wiley, 2015.