1.- DATOS DE LA ASIGNATURA

Nombre de la asignatura:	**Programación orientada a objetos**
Carrera:	**Ingeniería en Sistemas Computacionales**
Clave de la asignatura:	**SCM - 0426**
Horas teoría-horas práctica-créditos	**3-2-8**

2.- HISTORIA DEL PROGRAMA

Lugar y fecha de elaboración o revisión	Participantes	Observaciones (cambios y justificación)
Instituto Tecnológico de Toluca del 18 al 22 agosto 2003.	Representantes de la academia de sistemas y computación de los Institutos Tecnológicos.	Reunión nacional de evaluación curricular de la carrera de Ingeniería en Sistemas Computacionales.
Instituto Tecnológico de: Orizaba,Mexicali. 23 agosto al 7 de noviembre 2003.	Academia de sistemas y computación.	Análisis y enriquecimiento de las propuestas de los programas diseñados en la reunión nacional de evaluación.
Instituto Tecnológico de León 1 al 5 de marzo 2004.	Comité de consolidación de la carrera de Ingeniería en Sistemas Computacionales.	Definición de los programas de estudio de la carrera de Ingeniería en Sistemas Computacionales.

3.- UBICACIÓN DE LA ASIGNATURA

a). Relación con otras asignaturas del plan de estudio

Anteriores	
Asignaturas	**Temas**
Fundamentos de programación	Requiere del dominio de todos los temas para poder desarrollar modelos

Posteriores	
Asignaturas	**Temas**
Estructura de datos	

b). Aportación de la asignatura al perfil del egresado

Aplica la programación orientada a objetos en la solución de problemas reales que impliquen el desarrollo de software.

4.- OBJETIVO(S) GENERAL(ES) DEL CURSO

El estudiante aprenderá tópicos avanzados de programación orientada a objetos y su implementación por medio de un leguaje de programación, que sirvan como base para cursos posteriores donde se desarrollaran sistemas computacionales.

5.- TEMARIO

Unidad	Temas	Subtemas
1	Arreglos unidimensionales y multidimensionales.	1.1 Arreglo Unidimensionales listas (vectores). 1.1.1 Conceptos básicos. 1.1.2 Operaciones. 1.1.3 Aplicaciones. 1.2 Arreglo bidimensional. 1.2.1 Conceptos básicos. 1.2.2 Operaciones. 1.2.3 Aplicaciones. 1.3 Arreglo Multidimensional. 1.3.1 Conceptos básicos. 1.3.2 Operaciones. 1.3.3 Aplicaciones.
2	Métodos y mensajes.	2.1 Atributos **const** y **static**. 2.2 Concepto de método. 2.3 Declaración de métodos. 2.4 Llamadas a métodos (mensajes). 2.5 Tipos de métodos. 2.5.1 Métodos **const**, **static**. 2.5.2 Métodos normales y volátiles. 2.6 Referencia **this.** 2.7 Forma de pasar argumentos. 2.8 Devolver un valor desde un método. 2.9 Estructura del código.
3	Constructor, destructor.	3.1 Conceptos de métodos constructor y destructor. 3.2 Declaración de métodos constructor y destructor. 3.3 Aplicaciones de constructores y destructores. 3.4 Tipos de constructores y destructores.
4	Sobrecarga.	4.1 Conversión de tipos. 4.2 Sobrecarga de métodos. 4.3 Sobrecarga de operadores.

5.- TEMARIO (Continuación)

5	Herencia.	5.1 Introducción a la herencia. 5.2 Herencia simple. 5.3 Herencia múltiple. 5.4 Clase base y clase derivada. 5.4.1 Definición. 5.4.2 Declaración. 5.5 Parte protegida. 5.5.1 Propósito de la parte protegida. 5.6 Redefinición de los miembros de las clases derivadas. 5.7 Clases virtuales y visibilidad. 5.8 Constructores y destructores en clases derivadas. 5.9 Aplicaciones.
6	Polimorfismo y reutilización	6.1 Concepto del polimorfismo. 6.2 Clases abstractas. 6.2.1 Definición. 6.2.2 Redefinición. 6.3 Definición de una interfaz. 6.4 Implementación de la definición de una interfaz. 6.5 Reutilización de la definición de una interfaz. 6.6 Definición y creación de paquetes / librería. 6.7 Reutilización de las clases de un paquete / librería. 6.8 Clases genéricas (Plantillas).
7	Excepciones.	7.1 Definición. 7.1.1 Que son las excepciones. 7.1.2 Clases de excepciones, excepciones predefinidas por el lenguaje. 7.1.3 Propagación. 7.2 Gestión de excepciones. 7.2.1 Manejo de excepciones. 7.2.2 Lanzamiento de excepciones.

5.- TEMARIO (Continuación)

8	Flujos y archivos.	7.3 Excepciones definidas por el usuarios. 7.3.1 Clase base de las excepciones. 7.3.2 Creación de un clase derivada del tipo excepción. 7.3.3 Manejo de una excepción definida por el usuario. 8.1 Definición de Archivos de texto y archivos binarios. 8.2 Operaciones básicas en archivos texto y binario. 8.2.1 Crear. 8.2.2 Abrir. 8.2.3 Cerrar. 8.2.4 Lectura y escritura. 8.2.5 Recorrer. 8.3 Aplicaciones.

6.- APRENDIZAJES REQUERIDOS

- Comprender las estructuras de control de flujo.
- Crear y manipular datos primitivos.
- Contar con la capacidad de abstracción para analizar un problema y realizar el planteamiento de la solución mediante el uso de las técnicas básicas de análisis y diseño orientado a objetos.
- Tener la habilidad para desarrollar algoritmos que representen el comportamiento de los objetos involucrados en la solución del problema.
- Tener la habilidad de implementar el modelado obtenido para la solución de un problema, mediante una herramienta de desarrollo de software.

7.- SUGERENCIAS DIDÁCTICAS

- Uso de un portal de Internet para apoyo didáctico de la materia, el cual cuente por lo menos con un foro, preguntas frecuentes, material de apoyo y correo electrónico.
- Utilizar software didáctico y software de apoyo.
- Presentar proyectos finales
- Propiciar el uso de terminología técnica adecuada al programa.
- Definir los lineamientos de documentación que deberán contener las tareas y prácticas.
- Desarrollar de manera conjunta ejemplos de cada uno de los temas.

- Utilizar el aprendizaje basado en problemas, trabajando en grupos pequeños, para sintetizar y construir el conocimiento necesario para resolver problemas relacionados con situaciones reales.
- Solicitar al estudiante, la elaboración de los programas ejemplo en la computadora.
- Solicitar al estudiante propuestas de problemas a resolver y que sean significativas para él.
- Propiciar que el estudiante experimente con diferentes programas encontrados en revistas, Internet y libros de la especialidad, que lo lleven a descubrir nuevos conocimientos.
- Fomentar el trabajo en equipo.
- Elaborar de manera conjunta con el estudiante una guía de ejercicios para actividades extra clase
- Plantear problemas reales para que ellos los representen utilizando los conceptos de la POO.
- Uso del laboratorio para la elaboración de programas que integren los temas estudiados.
- Formar equipos de trabajo para la exposición de investigaciones y tareas
- Generar problemas prácticos y completos y solicitar la solución de aplicaciones utilizando la computadora
- Desarrollo de un proyecto con aplicación real.

8.- SUGERENCIAS DE EVALUACIÓN

- Participación y desempeño en el aula y el laboratorio.
- Dar seguimiento al desempeño en el desarrollo del programa (dominio de los conceptos, capacidad de la aplicación de los conocimientos en problemas reales, trasferencia del conocimiento).
- Desarrollo de un proyecto final que integre todas las unidades de aprendizaje.
- Participación del alumno en dinámicas grupales
- Actividades de auto evaluación.
- Exámenes departamentales.
- Cumplimiento de los objetivos y desempeño en las prácticas
- Exámenes escritos teórico - prácticos, exámenes en computadora.
- Programas asignados como tareas.
- Se recomienda utilizar varias técnicas de evaluación con un criterio de evaluación específico para cada una de ellas. (Se propone el criterio heurístico para los programas de cómputo desarrollados, axiológico para las prácticas grupales y criterio teórico para los exámenes de conocimiento. Los pesos que se le den a cada una de las técnicas se basara en la experiencia del profesor).

9.- UNIDADES DE APRENDIZAJE

UNIDAD 1.- Arreglos unidimensionales y multidimensionales.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
El estudiante conocerá la representación interna de los arreglos unidimensionales. Así mismo, será capaz de aplicarlos al construir modelos y desarrollar aplicaciones de software que requieran de estos.	1.1 Modelar objetos del mundo real que requieran de arreglos 1.2 Desarrollar los algoritmos de manipulación de los arreglos para realizar operaciones básicas. 1.3 Representar un arreglo por medio de una clase, que incluya los métodos que representan sus operaciones básicas. 1.4 Desarrollar un programa que implemente la clase –arreglo- y que interactué con otras clases	Todas

UNIDAD 2.- Métodos y mensajes.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Desarrollará clases que permitan implementar objetos que puedan comunicarse entre si por medio de mensajes parametrizados	2.1 Buscar la información sobre las diferentes formas en que puede implementarse un método. 2.2 Explicar los distintos tipos de parámetros que soportan los métodos. 2.3 Realizar una práctica donde se incluya el uso de mensajes entre objetos.	Todas

UNIDAD 3.- Constructor, destructor.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la función y las ventajas de los métodos constructores y destructores, y los aplicará en las clases	3.1 Buscar la información sobre las características de los métodos constructores y destructores. 3.2 Comparar las soluciones cuando se utilizan constructores y destructores contra soluciones donde no se utilizan. 3.3 Hacer una practica donde se apliquen constructores y destructores.	Todas

UNIDAD 4.- Sobrecarga..

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la filosofía, uso y la forma de implementar la sobrecarga tanto de métodos como de operadores.	4.1 Conocer el término de sobrecarga. 4.2 Explicar el funcionamiento de la sobrecarga de métodos. 4.3 Explicar el funcionamiento de la sobrecarga de operadores. 4.4 Hacer declaraciones de sobrecarga de métodos y operadores. 4.5 Desarrollar prácticas utilizando sobrecarga de métodos y operadores.	Todas

UNIDAD 5.- Herencia.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Comprenderá la filosofía, uso y la forma de implementar la sobrecarga tanto de métodos como de operadores.	5.1 Explicar cada uno de los conceptos utilizados en herencia. 5.2 Desarrollar un árbol de herencia. 5.3 Realizar programas completos utilizando los conceptos vistos en esta unidad. 5.4 Proporcionar ejercicios en donde se utilice la herencia	Todas

UNIDAD 6.- Polimorfismo y reutilización.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Desarrollará y utilizará interfaces, clases abstractas y paquetes/librerías para explotar la capacidad de reutilización de la POO.	6.1 Desarrollar una clase abstracta y crear subclases derivadas de ella. 6.2 Desarrollar una interfase y crear subclases derivadas de ella. 6.3 Desarrollar utilerías y encapsularlas para crear aplicaciones que la utilicen. 6.4 Realizar un reporte donde se comparen las clases abstractas y las interfaces.	Todas

UNIDAD 7.- Excepciones.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Identificará las condiciones de error que interrumpan el flujo normal de las sentencias en un programa y utilizará el marco controlador del manejo de excepciones para lograr desarrollar programas más seguros, estables y robustos	7.1 Buscar y seleccionar información referente al manejo de excepciones. 7.2 Analizar programas que no cuentan con manejo de excepciones y comparar resultados cuando se le agregan. 7.3 Realizar ejemplos para analizar el comportamiento en tiempo de ejecución de todas las variantes del manejo de excepciones.	Todas

UNIDAD 8.- Flujos y archivos.

Objetivo Educacional	Actividades de Aprendizaje	Fuentes de Información
Implementará aplicaciones orientadas a objetos que manipulen archivos de texto y binarios	8.1 Investigar los conceptos básicos de archivos. 8.2 Investigar las bibliotecas propias del lenguaje utilizado que sirven para interactuar con los archivos. 8.3 Proponer un caso de estudio que requiera el uso de archivos para que sea resuelto por el alumno.	Todas

10. FUENTES DE INFORMACIÓN

1. Taylor David.
   Object Orient informations systems, planning and implementations.
   Ed. Ed. Wiley, Canada, 1992.

2. Larman Craig.
   UML y patrones introducción al análisis y diseño orientado a objetos.
   Ed. Pretince Hall, México, 1999.

3. Winblad, Ann L. Edwards, Samuel R.
   Software orientado a objetos.
   Ed. Addison. Wesley/ Díaz Santos USA, 1993.

4. Deitel & Deitel.
   Java how to program.
   Ed. Prentice Hall.

5. Fco. Javier Ceballos.
   Java 2 Curso de Programación.
   Ed. Alfaomega.

6. Agustín Froufe.
   Java 2 Manual de usuario y tutorial.
   Ed. Alfaomega.

7. Laura Lemay, Rogers Cadenhead.
   Aprendiendo JAVA 2 en 21 días.
   Ed. Prentice Hall.

8. Herbert Schildt.
   Fundamentos de Programación en Java 2.
   Ed. McGrawHil.

9. J Deitel y Deitel.
   Como programar en Java.
   Ed. Prentice Hall.

10. Stephen R. Davis.
    Aprenda Java Ya.
    Ed. McGrawHill.

11. Kris Jamsa Ph D..
    ¡ Java Ahora!
    Ed. McGrawHill..

Referencias en Internet

[1] http:// www.javasoft.com
[2] http:// www.javaworld.com
[3] http:// www.prenhall.com/deitel

11. PRÁCTICAS

Unidad	Práctica	
1		
	1	El estudiante desarrollará la clase Array incluyendo todas las operaciones básicas que operan sobre un arreglo, tales como crear, insertar, eliminar, recorrer, buscar, modificar y destruir.
2	1	El estudiante comprenderá la forma de comunicación entre objetos de la misma clase o de diferente clase, donde por lo menos una de ellas se encuentra dentro de una librería o paquete de utilerías y ha sido definida por el alumno. Además, será capaz de enviar diferente tipos de mensajes.
3	1	El estudiante comprenderá los conceptos de los métodos constructor y destructor, además será capaz de implementarlos en la definición de una clase para optimizar el desempeño de la misma.
4	1	El estudiante comprenderá los conceptos de sobrecarga de operadores, además será capaz de implementarlos en la definición de una clase para optimizar el desempeño de la misma
5	1	El estudiante comprenderá los conceptos de la herencia, además será capaz de implementar clases subordinadas aplicando la sobrecarga incluyendo el método constructor y el destructor. Así como agregar nuevos métodos que agreguen funcionalidad de la clase heredada.
6	1	El estudiante creará una clase abstracta y derivar de ellas clases subordinadas. Además comprenderá las ventajas de implementar la abstracción.
7	1	Desarrollar un programa que favorezca la presencia de excepciones para que el alumno analice el comportamiento del manejo de excepciones por parte del sistema operativo.

Unidad	Práctica	
	2	Desarrollar un programa que permita la captura y manipulación de excepciones, en primera instancia sin distinción del tipo de excepción capturada, y como segunda instancia tomando distintas acciones según el tipo de excepción
	3	Desarrollar un programa que incluya el bloque de finalización.
	4	Desarrollar una aplicación que incluya por lo menos una clase excepción definida por el usuario, que la lance y la manipule
8	1	El estudiante desarrollará la clase TextFile incluyendo todas la operaciones básicas que operan sobre un archivo texto, tales como crear, abrir, cerrar, insertar, eliminar, recorrer, buscar, modificar y destruir.