Datos Generales			
Disciplina:	Técnicas de Programación de Computadoras		
Asignatura:	Programación 1		
Perfil:	**Ingeniero Informático.**		
Año:	1ro	**Semestre:** 2do	
Duración Total:	**90 horas**		

DISTRIBUCIÓN DE HORAS

Tema	C	CP	S	T	L	Eval	Total
Tema 1 Conceptos básicos del paradigma orientado a objetos	12	26			8	2	**48**
Tema 2 Temas complementarios para el desarrollo de aplicaciones informáticas	8	20			12	2	**42**
Totales	**20**	**46**			**20**	**4**	**90**

Objetivos Generales

- Caracterizar el paradigma de programación orientada a objetos a partir de la definición y el uso de sus conceptos fundamentales.
- Formalizar problemas computacionales mediante la modelación y aplicación de los conceptos de la programación orientada a objetos.
- Diseñar soluciones algorítmicas a partir de la formalización de un problema computacional usando los conceptos y buenas prácticas de diseño y programación, del paradigma orientado a objeto.
- Desarrollar aplicaciones informáticas de interfaz gráfica de usuario (GUI) mediante el uso de un entorno de desarrollo integrado, un lenguaje de programación orientado a objetos y el empleo de mecanismos que garanticen robustez, modularidad y persistencia de la información.

Contenido de la Asignatura

Sistema de conocimientos:

Concepto de objeto, clase e interfaz. Métodos formales de descripción de modelos orientados a objetos: Lenguaje UML. Mecanismos de encapsulación: interfaces de clases definidas por niveles de visibilidad y jerarquía de tipos. Polimorfismo universal: de tipo. Genericidad. Estructuras de datos básicas: colecciones mediante arreglos. Mecanismos de tratamiento de errores. Tratamiento de excepciones. Modelo orientado a objetos de excepciones. Fundamentos de diseño de interfaces gráficas de usuario. Eventos. Mecanismos para la persistencia de información.

Sistema de habilidades:

1. Diseñar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia y el polimorfismo.
2. Diseñar modelos mediante el uso de patrones y buenas prácticas de diseño del paradigma orientado a objeto.
3. Aplicar algoritmos básicos a la resolución de problemas computacionales, tales como algoritmos de búsqueda, conteo y ordenación, mediante el empleo de estructuras de datos predefinidas o modeladas según el paradigma orientado a objetos.
4. Implementar aplicaciones de interfaz gráfica de usuario, utilizando un entorno de desarrollo integrado, bibliotecas y marcos de trabajo.
5. Poner a punto aplicaciones informáticas de GUI mediante la validación de las operaciones que intervienen en el proceso de interacción con el usuario y el empleo de mecanismos para el tratamiento de excepciones.
6. Aplicar las bibliotecas de trabajo con ficheros en soluciones informáticas en las que sea necesario almacenar datos e información persistente.

Sistema de valores:

Deberán ser objeto de trabajo los siguientes valores:

- Honestidad.
- Sentido del trabajo.
- Responsabilidad.
- Solidaridad.
- Sentido de pertenencia.
- Creatividad.

Específicamente, se deberá puntualizar en los siguientes sentidos:

- Estimular el hábito de la autosuperación que se manifiesta en el estudio de la plataforma de trabajo y la búsqueda del conocimiento.
- Estimular el desarrollo de la creatividad y un enfoque independiente en la solución de diferentes problemas desde el punto de vista de la programación.
- Contribuir a formar profesionales en los que se conjuguen la alta calificación en el campo de estudios, con cualidades personales entre las que destaque la modestia y una actuación ética.

Descripción de los Temas

Tema I: Conceptos básicos del paradigma orientado a objetos

Objetivos:

- Caracterizar el paradigma de programación orientada a objetos a partir de la definición y uso de los conceptos de clase, herencia y polimorfismo.
- Modelar problemas computacionales mediante la aplicación de un lenguaje de modelado y los conceptos de clase, herencia y polimorfismo.
- Diseñar soluciones algorítmicas a partir de la modelación de un problema

computacional usando los conceptos, y buenas prácticas de diseño del paradigma orientado a objeto.

- Implementar soluciones algorítmicas a partir de la modelación de un problema computacional usando los conceptos y buenas prácticas de programación.

Sistema de conocimientos:

Concepto de objeto, clase e interfaz. Sobrecarga de métodos. Métodos formales de descripción de modelos orientados a objetos: Lenguaje UML. Mecanismos de encapsulación: interfaces de clases definidas por niveles de visibilidad y jerarquía de tipos. Herencia y Polimorfismo universal: de tipo.

Sistema de habilidades:

1. Caracterizar los conceptos de clase, agregación, composición, herencia y polimorfismo
2. Identificar clases y relaciones presentes en el texto de un problema.
3. Diseñar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia y polimorfismo, y las buenas prácticas de diseño del paradigma orientado a objeto.
4. Implementar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia y polimorfismo, y las buenas prácticas de programación del paradigma orientado a objeto.
5. Aplicar algoritmos básicos, tales como algoritmos de búsqueda, conteo y ordenación en la solución de un problema computacional.

Evaluación del Tema:

Evaluaciones frecuentes, 1ra Prueba Parcial, y Examen Final.

Tema II: Temas complementarios para el desarrollo de aplicaciones informáticas

Objetivos:

- Desarrollar aplicaciones informáticas de GUI mediante la validación de las

operaciones que intervienen en el proceso de interacción con el usuario y el mecanismo para el tratamiento de excepciones para crear aplicaciones robustas.

- Utilizar la Genericidad como recurso que permite crear estructuras de datos reutilizables para facilitar el desarrollo de aplicaciones.
- Desarrollar aplicaciones informáticas de GUI que requieran el uso de ficheros para almacenar información persistente.

Sistema de conocimientos:

Genericidad. Estructuras de datos básicas: colecciones mediante arreglos. Mecanismos de tratamiento de errores. Tratamiento de excepciones. Modelo orientado a objetos de excepciones. Bibliotecas de interfaces gráficas de usuario. Eventos. Mecanismos para la persistencia de información.

Sistema de habilidades:

1. Implementar aplicaciones de interfaz gráfica de usuario, utilizando un entorno de desarrollo integrado, bibliotecas y marcos de trabajo.
2. Validar las operaciones que intervienen en el proceso de interacción con el usuario mediante el tratamiento de errores.
3. Utilizar los mecanismos para el tratamiento de excepciones.
4. Aplicar la genericidad para diseñar, implementar y usar colecciones mediante arreglos.
5. Aplicar las bibliotecas de trabajo con ficheros en soluciones informáticas en las que sea necesario almacenar datos e información persistente.

Evaluación del Tema:

Evaluaciones frecuentes, 2da Prueba Parcial y Examen Final.

Indicaciones Metodológicas y de organización de la asignatura

Se cuenta con un sistema de conferencias, clases prácticas y laboratorios por cada

nuevo contenido.

Las clases prácticas han de dedicarse al análisis y desarrollo de soluciones, sobre todo desde el punto de vista del diseño y desde el punto de vista algorítmico; deben proveer el desarrollo de ejercicios que solucionen problemas de baja y mediana complejidad. En cada clase práctica la habilidad primaria es la de diseño e implementación, se recomienda el trabajo con situaciones de un solo contexto, donde el estudiante pueda desarrollar varios tipos de algoritmos usando el mismo dominio de aplicación.

Los laboratorios deben ser un mecanismo de estimulación al estudio individual y a la experimentación. Son actividades donde la labor del estudiante ha de desarrollarse en la computadora en un 100%, sobre todo usando el entorno de desarrollo. En estas actividades de laboratorio, ha de hacerse énfasis en el trabajo con el IDE, el trabajo con los componentes visuales y la puesta a punto de aplicaciones informáticas. El profesor debe facilitar al estudiante, en el caso que lo requieran, los medios necesarios para desarrollar el laboratorio en tiempo, siempre y cuando estos medios no entren en contradicción con el objetivo del laboratorio. Estos medios pueden ser clases predefinidas, interfaz visual construida con las ventanas necesarias para la entrada y salida de datos, declaraciones de clases y métodos, estructuras de datos con valores iniciales para realizar las pruebas, etc.

El estudio y el trabajo independiente deben ser actividades dirigidas a que el estudiante sistematice las habilidades formadas en clases, que le permitan alcanzar un nivel de asimilación productivo y creativo en la resolución de problemas que requieran y estimulen a la investigación, a la consulta de literatura científica en idioma nacional e inglés referente a los aspectos de la programación orientada a objetos, y a la búsqueda de nuevas vías y estrategias de solución de problemas.

Sistema de evaluación de la Asignatura

La evaluación de la asignatura se realizará a partir de la asistencia y participación en clases, el resultado de las evaluaciones frecuentes, las dos pruebas parciales y el examen final.

La nota final estará determinada por el avance (Trayectoria) de cada estudiante, dado el conjunto de resultados logrados durante el curso y el resultado del Examen Final. Cada evaluación debe ser calificada de manera cualitativa con las notas 2 (Mal), 3 (Regular), 4 (Bien) y 5 (Excelente).

El sistema de evaluación presenta dos pruebas parciales.

Objetivos de la 1ra Prueba Parcial

1. Caracterizar los conceptos de clase, agregación, composición, herencia y polimorfismo
2. Identificar clases y relaciones presentes en el texto de un problema.
3. Diseñar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia, y las buenas prácticas de diseño del paradigma orientado a objeto.
4. Implementar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia, y las buenas prácticas de programación del paradigma orientado a objeto.
5. Aplicar algoritmos básicos, tales como algoritmos de búsqueda, conteo y ordenación en la solución de un problema computacional.

Objetivos de la 2da Prueba Parcial

1. Implementar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia, y las buenas prácticas de programación del paradigma orientado a objeto.
2. Aplicar algoritmos básicos, tales como algoritmos de búsqueda, conteo y ordenación en la solución de un problema computacional.
3. Validar las operaciones que intervienen en el proceso de interacción con el usuario mediante el tratamiento de errores.
4. Utilizar los mecanismos para el tratamiento de excepciones.

La **prueba final** tiene los siguientes objetivos

1. Diseñar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia y polimorfismo, el empleo de buenas prácticas de diseño del paradigma orientado a objeto.
2. Implementar clases y relaciones presentes en el planteamiento de un problema aplicando los conceptos de encapsulación, herencia y polimorfismo, y el empleo de buenas prácticas de programación del paradigma orientado a

objeto.

3. Aplicar algoritmos básicos, tales como algoritmos de búsqueda, conteo y ordenación en la solución de un problema computacional.
4. Utilizar los mecanismos para el tratamiento de excepciones.
5. Aplicar la genericidad para diseñar, implementar y usar colecciones mediante arreglos.
6. Aplicar las bibliotecas de trabajo con ficheros de texto en soluciones algorítmicas en las que sea necesario almacenar y extraer datos e información persistente.

Bibliografía

Básica:

1. Análisis y diseño orientado a objetos con aplicaciones. Grady Booch, Ivar Jacobson, y James Rumbaugh, Series Editors. The Addison-Wesley Object Technology Series. 2007.(Formato digital)
2. Object-Oriented Programming. Timothy Budd. Addison-Wesley Iberoamericana 1994. (Libro de texto)
3. Cómo programar en C/C++. H.M. Deitel y P.J. Deitel.(Libro de texto)
4. Programación en Java 2 J2SE 1.4. John Zukowski.(Libro de texto)
5. Microsoft C# Professional Projects. G. Arora, B. Aiaswamy y N. Pandey.(Libro de texto)

Complementaria:

1. Manuales digitales de la asignatura. (Formato digital)
2. Touch of Class. Meyer, Bertrand. 2008.(Formato digital)
3. Ayuda en línea del ambiente de programación utilizado durante el curso.

Reelaborado por:

Msc. Yaniela Fernández Mena Asesora DDM Técnicas de Programación

Fecha: Enero / 2014