Programa Analítico

1. OBJETIVOS:

La asignatura comprende dos grandes áreas temáticas: a) programación orientada a objetos y b) programación orientada a procesos. Ambas partes son independientes, por lo tanto pueden ser enseñadas en forma paralela o en forma consecutiva sin importar el orden de las mismas. A continuación se describen los objetivos de cada área.

A. Adquirir conceptos avanzados de la programación orientada a objetos utilizando como lenguaje de programación JAVA. En base a este objetivo principal, se establecen los siguientes objetivos particulares:

   - Tener claros cuáles son los conceptos básicos en torno a los que gira la programación orientada a objetos y llegar a comprender el porqué de las relaciones entre dichos conceptos.
   - Adquirir la mentalidad de elaboración de abstracciones necesaria para elaborar estructuras de programación.
   - Familiarizarse con los términos técnicos, como polimorfismo, herencia, interfaz y clases abstractas y comprender los conceptos a los que se refieren.
   - Saber cuáles son las metodologías que se emplean para llevar adelante un proyecto de programación y estar en disposición de desarrollar estrategias eficientes.
   - Aprender las ventajas de la herencia a la hora de desarrollar objetos complejos.
   - Conocer las ventajas de la reutilización de código y qué pasos hay que seguir para elaborar arquitecturas de programación que faciliten al máximo esta reutilización.

B. Adquirir conocimientos necesarios para la programación de las funciones utilizadas frecuentemente en aplicaciones como editores de texto, compiladores o sistemas operativos utilizando como lenguaje de programación ANSI C. En base a este objetivo principal, se establecen los siguientes objetivos particulares:

   - Aprender el uso de punteros.
   - Familiarizarse con la técnica de recursión y conocer sus ventajas y desventajs frente a técnicas iterativas.
   - Aprender el procesamiento de cadenas.

2. CONTENIDOS:

Parte A

UNIDAD 1. Diseño de clases

1.1 - Acoplamiento y cohesión. Encapsulamiento.
1.2 - Prueba y depuración de programas: pruebas de unidad, pruebas automatizadas, seguimiento manual, depuradores. Pruebas positivas y negativas.
1.3 - Modularización e interfaces. Comentarios y estilos. Paquetes. Interfaz de usuario.

UNIDAD 2. Organización de clases

2.1 - Herencia. Jerarquías de herencia. Superclases y subclases. Derechos de accesos. Uso de constructores heredados. La palabra reservada super. Subclases y asignación.
2.2 - Enmascaramiento de tipos. La clase Object y las clases "envoltorios": Integer, Character, Double, etc.

UNIDAD 3. Polimorfismo

3.1 - Tipo estático y dinámico.
3.2 - Sobreescritura de atributos y métodos.
3.3 - Clases y métodos abstractos.
3.4 - Diseño e implementación de interfaces.
3.5 - Diferencia entre clases abstractas e interfaces.
3.6 - Herencia simple y herencia múltiple

UNIDAD 4. Excepciones

4.1 - Definición de excepción. Uso de las excepciones predefinidas.
4.2 - Captura de excepciones.
4.3 - Lanzamiento de excepciones.
4.4 - Definición de excepciones de usuario.
4.5 - Propagación de excepciones.
4.6 - La cláusula finally.

UNIDAD 5. Archivos

5.1 - Definición y clasificación.
5.2 - Archivos de texto. Procesamiento de archivos de texto.
5.3 - Serialización de objetos. Escribir objetos en un archivo y leer objetos desde un archivo.
5.4 - Archivos binarios. Procesamiento de archivos binarios.

UNIDAD 6. Interfaz gráfica de usuario

6.1 - Programación con interfaz gráfica de usuario.
6.2 - Applets y aplicaciones gráficas. Diferencias. Ciclo de vida de applets.
6.3 - Programación dirigida por eventos. Objetos gráficos: botones de comando, bordes, cajas de texto, etiquetas, botones de opciones y de verificación, cajas de listas, barras de desplazamiento, etc. Manejo de eventos sobre los objetos gráficos

Parte B

UNIDAD 7. Programación imperativa.

7.1 - Estructuras de control.
7.2 - Representación en memoria de enteros, reales y caracteres.
7.3 - Tipos de datos primitivos.Tamaño. Constantes. Especificación de constantes enteras en octal y hexadecimal. Secuencias de escape. Declaración de variables.
7.4 - Operadores aritméticos, de relación, lógicos y de manejo de bits. Operador condicional.
7.5 - Conversiones de tipo. Precedencia y orden de evaluación.

UNIDAD 8. Arreglos

8.1 - Declaración de un arreglo lineal. Almacenamiento en memoria.
8.2 - Asignación de valores.
8.3 - Procesamiento de arreglos lineales: recoorrer un arreglo; buscar el mínimo o máximo; ordenar sus elementos.
8.4 - Arreglos bidimensionales. Almacenacenamiento en memoria.
8.5 - Programación de operaciones matriciales: suma y producto de matrices; producto escalar, etc.

UNIDAD 9. Funciones

9.1 - Deficinición de una función.
9.2 - Parámetros y valores de retorno.
9.3 - Pasaje de parámetros por valor.
9.4 - Funciones recursivas.

UNIDAD 10. Punteros y procesamiento de cadenas

10.1 - Punteros. Punteros y direcciones.
10.2 - Pasaje de los parámetros de una función por dirección.
10.3 - Punteros y arreglos.
10.4 - Aritmética de punteros.
10.5 - Punteros a caracteres y a funciones. Procesamiento de cadenas.
10.6 - Punteros a punteros.
10.7 - Punteros y arreglos multidimensionales.

UNIDAD 11. Estructuras

11.1 - Concepto. Estructuras y funciones.
11.2 - Arreglos de estructuras.
11.3 - Punteros a estructuras.
11.4 - Aplicaciones: búsqueda en tablas.

3. BIBLIOGRAFIA

3.1 BASICA

1. **Barnes, Kölling**. *Programación Orientada A Objetos Con Java. Una Introducción Práctica Usando Bluej*. Pearson Educación, S.A. -2007

2. **Kernighan, Ritchie**. *El Lenguaje De Programación C.* Segunda Edición. Prentice Hall Hispanoamericana S.A. – 1991

3.2 Adicional

1. **Cohoon, Davidson. Programación En Java 5.0. Mcgraw-Hill. 2006**

4.METODOLOGIA DE LA ENSEÑANZA

El desarrollo del curso se compone de clases teóricas, clases de resolución de problemas y practicas de programación, estas últimas se desarrollarán en el laboratorio de informática.

En estas clases el alumno deberá realizar aplicaciones de software utilizando los conceptos dados en la clase teórica.

Las guías de ejercicios, enunciados de trabajos prácticos y apuntes de interés se publican en el Portal de la Universidad.

5. CRITERIOS DE EVALUACION

Además de los exámenes parciales dispuestos por la Universidad se evaluará a los alumnos a través de pruebas semanales o quincenales y entrega de trabajos.

Para la aprobación de la cursada se tendrá en cuenta los siguientes conceptos:

- Calificación obtenida en los exámenes parciales.
- Evaluación de los trabajos prácticos realizados (individuales o grupales).

Lo anterior junto con

- El resultado de las evaluaciones breves que los profesores toman como seguimiento.
- Asistencia, puntualidad, respeto y participación en clase.

Conformarán una nota de concepto.

Para la aprobación de la materia el alumno deberá rendir un examen final que consiste en la resolución de diferentes problemas utilizando el lenguaje de programación utilizado durante el dictado del curso. La calificación final estará conformada por la nota del examen y la nota de concepto.