PROGRAMA de TÉCNICAS AVANZADAS DE PROGRAMACIÓN[1]

Carrera: Ingeniería en Automatización y Control Industrial.

Asignatura: Técnicas Avanzadas de Programación[2]

Núcleo al que pertenece: Núcleo Superior Básico[3]

Profesoras/es: Eric Pernia

Asignaturas previas necesarias para favorecer el aprendizaje: Organización y Arquitectura de Computadores.

Objetivos:

- Comprender el paradigma de Programación Orientado a Objetos y sus conceptos básicos: objeto, mensaje, polimorfismo, colecciones, estado.
- Ser capaz de realizar un diseño e implementación de software usando este paradigma. En particular, maneje una correcta distribución de responsabilidades, conozca formas de reutilización de código; y entienda los conceptos de colaboración, delegación y encapsulamiento.
- Obtener nociones de arquitecturas de software, incluyendo programación de interfaces y persistencia.

Contenidos mínimos:

El Paradigma de Programación orientado a objetos (POO): objeto, mensaje, polimorfismo, colecciones, estado. Efecto de lado. Diseño de software usando este paradigma POO: Distribución de responsabilidades. Formas de reutilización de código. Colaboración. Delegación. Encapsulamiento. Nociones de arquitecturas de software: Patrón Modelo Vista Controlador (MVC).

Carga horaria semanal: *6 horas*

---

1 *Fecha de última modificación de este documento: 2021-07-19.*

2 *En el Plan vigente, ResCS Nº455-15. Para el Plan RCS Nº 183-03 s*e llama C*omputadores II.*

3 E*n el Plan vigente, ResCS Nº455-15. Para el Plan RCS Nº 183-03 pertenece al Núcleo Básico del Ciclo Superior.*

Programa analítico:

Unidad 1: Paradigma de Programación orientada a objetos.

- Paradigma de programación como marco conceptual a partir del cual se piensa el software y su construcción, existencia de distintos paradigmas.
- Conceptos fundantes de la programación con objetos: objeto y mensaje. Organización de un programa a partir de la definición de objeto y la asignación de comportamiento a los mismos mediante la identificación y asignación de responsabilidades.
- Polimorfismo entre objetos: concepto, aplicabilidad en casos sencillos. Su rol fundamental para construir software confiable y componentes con alto grado de genericidad.
- Implementación del comportamiento de un objeto: estado interno, método.
- Objetos básicos: números, strings, booleanos. Manejo de conjuntos y listas mediante objetos ad-hoc: colecciones, su comportamiento, formas de interacción.
- Implementación de estructuras de control como comportamiento de objetos básicos.
- Clases: repositorio de definiciones y molde de objetos con comportamiento similar.
- Lenguajes que soportan la programación con objetos: Smalltalk, Java, C\#, Eiffel, etc. Comparación, fortalezas y debilidades relativas. Posibilidad de utilizar conceptos de programación con objetos en otros ambientes, p.ej. Mathlab.

Unidad 2: Diseño de software usando objetos.

- Construcción de software a partir de conceptos que abstraen las características del hardware subyacente: características, aplicabilidad, fortalezas y debilidades.
- Utilización de las clases para factorizar las definiciones necesarias. La herencia como relación entre clases y como mecanismo para profundizar esta factorización. Modelado de alternativa mediante herencia, template method.
- Relaciones entre objetos: conocimiento, composición, agregación. Breve introducción a los diagramas UML. Distinción entre relaciones entre objetos y relaciones entre clases.

- Tipos: concepto de tipo en la programación con objeto, distinción entre tipo y clase, aplicación del concepto de protocolo.
- Chequeo de tipos: distinción entre el concepto de tipo y la posibilidad de chequeo, chequeos en tiempo de compilación o de ejecución, formalización de los protocolos.
- Qué es el diseño con objetos. Técnicas básicas: identificación de responsabilidades, definición de objetos candidatos a partir del comportamiento, conceptualización de protocolos útiles, interacción dinámica entre la definición del comportamiento de los objetos detectados y su implementación. Desarrollo iterativo de software.
- Utilización de los conceptos y técnicas de diseño en la construcción de software de control, modelado de componentes mediante objetos adecuados, aprovechamiento del polimorfismo y del concepto de protocolo.
- Límites de la herencia, problemas cuya resolución utilizando herencia resulta deficiente. Distintas formas de definir y manejar la composición entre objetos. La idea de design pattern. Algunos patterns de uso extendido: strategy, state, composite, observer, command.

Unidad 3: Nociones de arquitecturas de software.

- Programación de interfaces.
    - Manejo de flujos de datos entrantes y salientes: streams, modelado mediante protocolos ad-hoc, objetos que implementan estos protocolos.
    - Interfaz de usuario: objetos que modelan la información que se despliega en pantalla y la ingresada por los usuarios. Técnicas básicas de construcción de interfaces de usuario, incorporación de los datos ingresados a un modelo de objetos. Aplicabilidad de técnicas de diseño de software y de manejo de errores.
- Persistencia.
    - Estrategias de Persistencia mediante base de datos relacionales.
    - Operaciones de manipulación de datos en una base de datos relacional. Estrategias de acceso a los datos mediante consultas, concepto de índice.
    - Conceptos de usuario y permiso en una base de datos.

- Transacciones de base de datos.
- Acceso desde una aplicación.

Bibliografía obligatoria:

- Budd, Timothy A. An introduction to Object-Oriented programming. 2004.
- Booch Grady. Análisis y diseño orientado a objetos con aplicaciones, Editorial Addison-Wesley Iberoa (2da edición).
- Alex Sharp. Smalltalk by example, McGraw Hill, 1997.
- Gamma, Helm, Johnson, Vlissides. Design Patterns. Elements of Reusable Objects Oriented Software, Addison-Wesley Professional Computing Series.
- Rumbaugh, Jacobson and Booch. The UML Reference Manual, Addison Wesley Longman, Inc, 1998.

Bibliografía de consulta:

- Apuntes de Wollok: https://www.wollok.org/documentacion/apuntes/

Modalidad de evaluación

Régimen regular

Para la evaluación de los conocimientos de los alumnos, habrá ejercicios de entrega obligatoria, un trabajo práctico grupal de entrega obligatoria (con entregas incrementales) y una evaluación integradora al terminar el cuatrimestre.

La ponderación de las notas para la nota final será:

Actividad evaluadora	% de nota
Ejercicios de entrega obligatoria	10 %
Trabajo práctico grupal obligatorio	20 %
Evaluación integradora	70 %

Exámenes libres:

Deberá rendir un examen teórico-práctico, con las herramientas de software del curso que deberá consultar con el docente a cargo al realizar la inscripción al libre.

Anexo I

CRONOGRAMA TENTATIVO

Semana	Tema/Unidad	Actividad				Evaluación
		Teórico	Práctico			
			Res. Prob	Lab.	Otros Espec.	
1	Paradigma de programación como marco conceptual a partir del cual se piensa el software y su construcción, existencia de distintos paradigmas. Conceptos fundantes de la programación con objetos: objeto y mensaje. Organización de un programa a partir de la definición de objeto y la asignación de comportamiento a los mismos mediante la identificación y asignación de responsabilidades.	x	x			
2	Polimorfismo entre objetos: concepto, aplicabilidad en casos sencillos. Su rol fundamental para construir software confiable y componentes con alto grado de genericidad. Implementación del comportamiento de un objeto: estado interno, método.	x	x			
3	Objetos básicos: números, strings, booleanos. Manejo de conjuntos y listas mediante objetos ad-hoc: colecciones, su comportamiento, formas de interacción. Implementación de estructuras de control como comportamiento de objetos básicos.	x	x			x
4	Clases: repositorio de definiciones y molde de objetos con comportamiento similar. Lenguajes que soportan la programación con objetos: Smalltalk, Java, C\#, Eiffel, etc. Comparación, fortalezas y debilidades relativas. Posibilidad de utilizar conceptos de programación con objetos en otros ambientes, p.ej. Mathlab.	x	x			
5	Construcción de software a partir de conceptos que abstraen las características del hardware subyacente: características, aplicabilidad, fortalezas y debilidades. Utilización de las clases para factorizar las definiciones necesarias. La herencia como relación entre clases y como mecanismo para profundizar esta factorización. Modelado de alternativa mediante herencia, template method.	x	x			x
6	Relaciones entre objetos: conocimiento, composición, agregación. Breve introducción a los diagramas UML. Distinción entre relaciones entre objetos y relaciones	x	x			

	entre clases. Tipos: concepto de tipo en la programación con objeto, distinción entre tipo y clase, aplicación del concepto de protocolo.					
7	Chequeo de tipos: distinción entre el concepto de tipo y la posibilidad de chequeo, chequeos en tiempo de compilación o de ejecución, formalización de los protocolos. Qué es el diseño con objetos. Técnicas básicas: identificación de responsabilidades, definición de objetos candidatos a partir del comportamiento, conceptualización de protocolos útiles, interacción dinámica entre la definición del comportamiento de los objetos detectados y su implementación. Desarrollo iterativo de software.	x	x			
8	Utilización de los conceptos y técnicas de diseño en la construcción de software de control, modelado de componentes mediante objetos adecuados, aprovechamiento del polimorfismo y del concepto de protocolo. Límites de la herencia, problemas cuya resolución utilizando herencia resulta deficiente. Distintas formas de definir y manejar la composición entre objetos. La idea de design pattern. Algunos patterns de uso extendido: strategy, state, composite, observer, command.	x	x			x
9	Programación de interfaces. Persistencia.	x	x			
10	Evaluación integradora.	x	x			X
11	Seguimiento de trabajos grupales.		x			
12	Seguimiento de trabajos grupales.		x			
13	Seguimiento de trabajos grupales.		x			
14	Seguimiento de trabajos grupales.		x			
15	Seguimiento final y entrega de trabajos grupales.		x			x
16	Recuperatorio de evaluación integradora.	x	x			x
17	Actividad retrospectiva.				Dinámica	

*indique con una cruz la modalidad

Firmado digitalmente por: SUAREZ Mariana Alejandra
Directora del Departamento de Ciencia y Tecnología
UNIVERSIDAD NACIONAL DE QUILMES