



**Tecnicatura Superior Análisis,
Desarrollo y Programación de
Aplicaciones**

Plan 2003 -Res: 6175/03

PROGRAMA DE LA MATERIA:	11213 Programación Orientada a Objetos
-------------------------	---

Area: Sistemas	Equipo Docente: Cherencio, Guillermo
Cantidad de horas semanales : 4 hs.	
Días de dictado:	Segundo Año
L M M J V S	
2 2	

Materias Antecedentes	Materias Consecuentes
11106 Programación I	11321 Práctica Profesional

1.0 Objetivos, Expectativas de Logro

Que el alumno sea capaz de:

1. Aplicar diferentes estructuras de datos y de control adecuándolas a cada problema particular.
2. Diseñar algoritmos de mediana y alta complejidad
3. Programar aplicando el paradigma de programación orientada a objetos
4. Resolver problemas a través de un diseño orientado a objetos
5. Implementar un diseño orientado a objetos utilizando un lenguaje orientado a objetos
6. Representar un diseño orientado a objetos utilizando UML.

Contenidos mínimos:

1. Paradigma de Objetos: Conceptos de Objeto, Clase, Encapsulación, Abstracción, Jerarquía, Modularidad, Herencia, Polimorfismo.
2. Control, propiedades, métodos, eventos.
3. Creación de ejecutables, librerías dinámicas y estáticas.
4. Acceso a datos, técnicas de acceso y control.
5. Implementación de Altas, Bajas, Modificaciones y Consultas. Gestión de controles de usuario.
6. Documentación y mantenimiento de programas.

Abril 2021

2.0 Programa de Estudio

1.0 Modularidad y Reusabilidad

- 1.1 Repaso de la unidad análoga de Programación I.
- 1.2 Repaso: Descomposición de problemas.
- 1.3 Repaso: Funciones, declaración, implementación, uso. Enfoque Top-Down versus Down-Top. Reutilización de código.
- 1.4 Bibliotecas. Biblioteca standard.
- 1.5 Bibliotecas Estáticas y Dinámicas.
- 1.6 Repaso: Compilación, Linkedición, Ejecución.
- 1.7 Scripts para creación de ejecutables. Combinación de código para crear un ejecutable.
- 1.8 Compilación de librerías estáticas y dinámicas.
- 1.9 TP I: Creación de Ejecutables y Librerías Dinámicas.

2.0 Programación Orientada a Objetos I Objetos

- 2.1 Introducción. La programación orientada a objetos Vs. La programación tradicional.
- 2.2 Un progreso en el nivel de abstracción.
- 2.3 Objetos. Atributos. Métodos. Estado. Identificación. Intercambiables.
- 2.4 Encapsulación. Modificadores de Accesos.
- 2.5 Representación de Objetos y otros conceptos POO: UML. Definición. Tipos de Diagramas. Ejemplos.
- 2.6 Utilización de objetos. Instanciación. Ejemplo en Java.
- 2.7 Vínculos entre objetos, colaboración, navegabilidad.
- 2.8 Mensajes y Operaciones entre Objetos.
- 2.9 Asociación. Agregación. Diferencias. Ejemplos. Diagramas UML de Asociación y Agregación.
- 2.10 Implementación de objetos: Garbage Collector. Método `finalize`. Ciclo de Vida de los objetos.
- 2.11 Lenguajes Orientados a Objetos: Smalltalk, Eiffel, C++, Java.
- 2.12 La plataforma Java. Lenguaje Compilado e Interpretado. Portabilidad: "Write Once, Run Anywhere". Arquitectura. Plataformas: J2SE, J2ME, J2EE. J2SDK y J2RE.
- 2.13 TP II: Instalación y Uso de Plataforma de Desarrollo (J2SE SDK, J2SE SDK Docs, J2SE Tutorial, BlueJ IDE)
- 2.14 TP IIb: Uso de Objetos.

3.0 Programación Orientada a Objetos II Clases

- 3.1 Clases, definición, diseño de objetos.
- 3.2 Datos y Operaciones compartidas.
- 3.3 Atributos: tipos, tipos de accesos. Reglas de Encapsulación. Modificadores de Accesos.
- 3.4 Operaciones: Constructores, Setters, Getters, Customs.
- 3.5 Tipos de Datos vs Clases.
- 3.6 Reutilización de código a través del uso de Clases.
- 3.7 Diferentes roles: "class creator" vs "client programmer".

- 3.8 Diagramas de Clases.
- 3.9 Creación de una Clase. Ejemplo en Java.
- 3.10 TP III: Implementación de Clase.
- 4.0 Programación Orientada a Objetos III Elementos de los Lenguajes OO**
 - 4.1 Elementos de los lenguajes: Palabras Reservadas, Identificadores, uso de signos (" ; , { }"), espacios en blanco, etc.
 - 4.2 Tipos de datos primitivos y referencias, requerimiento de almacenamiento.
 - 4.3 Modelo de Memoria: Tipos de Datos (Primitivos y Referencias), Stack, Heap, Variables, Constantes, Almacenamiento static, no static, Objetos, datos de instancia, datos de clase, datos locales a métodos.
 - 4.4 Creación de objetos y memoria: usando constructores, usando otras clases.
 - 4.5 Mutabilidad: Clases mutables e inmutables, cualificador `final`.
 - 4.6 Limpieza de memoria: Garbage Collector, método `finalize`.
 - 4.7 Operadores. Operadores de Objetos: `new`, el operador de punto. Operadores: asignación, aritméticos, precedencia y asociación, booleanos, condicionales, de bits, etc.
 - 4.8 Casting y conversión sobre tipos de datos primitivos.
 - 4.9 Tipos de datos primitivos y objetos: las clases Wrappers.
 - 4.10 Tipo de dato caracter. Clases `String` y `StringBuffer`.
 - 4.11 Otras operaciones aritméticas: la Clase `Math`.
 - 4.12 Dentro de los métodos: estructuras de control. Secuencia. Selección (`if`, `switch`). Iteración: inicialización, prueba, ejecución, alteración de condición (`do - while`, `for`, `break`, `continue`, `label`).
 - 4.13 Entrada/Salida con y sin formato: la clase `System`.
 - 4.14 Documentación: propósito. ¿Cómo documentar clases?. Uso de comentarios y herramienta `javadoc`. Parámetros, `javadoc` y `html`.
 - 4.15 TP IV: Implementación de métodos de Clase y documentación.
- 5.0 Programación Orientada a Objetos IV Herencia**
 - 5.1 Herencia. Herencia Múltiple. El problema de la Herencia Múltiple. ¿Qué se hereda?.
 - 5.2 Diseñando una jerarquía de clases.
 - 5.3 Clase Base, Padre o Superclase. Clases Derivada, Subclase, Hija.
 - 5.4 Atributos: Herencia y Modificadores de Accesos.
 - 5.5 Operaciones: signatura o firma de métodos.
 - 5.6 Agregando nuevos comportamientos a clases derivadas.
 - 5.7 Modificando comportamientos en clases derivadas.
 - 5.8 Sobrecarga y Sobreescritura (Overloading vs Overriding).
 - 5.9 Clases vs Tipos de Datos II: Upcasting Vs Downcasting.
 - 5.10 Implementación de la Clase `Stack` usando herencia.
 - 5.11 Implementación de la Clase `Stack` usando composición.
 - 5.12 Herencia Vs Composición. Ventajas y Desventajas.
 - 5.13 Herencia Vs Asociación. Relaciones "IS A" vs "HAS A"

- 5.14 Tipos de Clases: Concretas y Abstractas.
- 5.15 Clases final y herencia.
- 5.16 Derivando de Clases Abstractas. Implementando métodos abstractos.
- 5.17 Implementación de Herencia: Instanciando objetos en una jerarquía de clases. Ejecución de constructores. Ejemplo en Java.
- 5.18 TP V: Implementación de Herencia.
- 6.0 Programación Orientada a Objetos V Interfases, Polimorfismo, Packages, Excepciones**
 - 6.1 Más allá de las Clases Abstractas: Interfases. Uso, implementación.
 - 6.2 Tipos estáticos y tipos dinámicos.
 - 6.3 Variables polimórficas.
 - 6.4 Mensajes polimórficos.
 - 6.5 Dynamic Binding: conectando un mensaje a un método en runtime. Ejemplo en Java.
 - 6.6 Casting: Explícito e Implícito. Ejemplo en Java.
 - 6.7 Clases Genéricas. Implementación.
 - 6.8 ¿Cómo agrupar clases? Framework. Definición. Ejemplos. Packages, carga de clases, efectos de los packages en la POO, packaging, packages y deployment, distribución de packages.
 - 6.9 Manejo de excepciones en los lenguajes tradicionales. Tipos de errores. Lanzando (throwing) y manipulando (handling) excepciones.
 - 6.10 Jerarquía de clases asociadas con excepciones, las clases: Throwable, Error, Exception, RuntimeException.
 - 6.11 Lidando con Excepciones: capturando excepciones, declarando y manipulando, bloques try - catch - finally.
 - 6.12 Excepciones y control de flujo. Rethrowing. Ejemplos. Ejercitación.
 - 6.13 Creando nuestras propias excepciones. Relgas de sobreescritura (overriding) vinculadas con excepciones.
 - 6.14 TP VI: Polimorfismo, Packaging, Interfases, Excepciones.
- 7.0 Programación Orientada a Objetos VI GUI's y POO**
 - 7.1 GUI. Definición. Aplicaciones. Diseño de GUI's.
 - 7.2 El framework AWT.
 - 7.3 Pasos para crear una GUI: Diseñando clases, creando componentes (controles), seleccionando contenedores, seleccionando layout managers, dando tamaño a componentes y contenedores, mostrando la GUI.
 - 7.4 Modelo de Eventos. Objetos de Eventos. Clases Listener. Clases Adapter. Inner Clases. Clases Anónimas.
 - 7.5 Patrones de Diseño. El patrón Model View Controller Pattern. Implementación. Ejercitación.
 - 7.6 El framework Swing. Relación con AWT. Componentes livianos vs pesados.
 - 7.7 JFC. Swing GUI Components: Contenedores y Componentes. Soporte para Drag & Drop, Cut, Copy, Paste, HTML. Java Tutorial: guías "How To Use..."
 - 7.8 Contenedores. Contenedores de Alto Nivel (JFrame, JApplet, JDialog),

- De Propósito General (JPanel, JScrollPane, JSplitPane, JTabbedPane, JToolBar), De Propósito Especial (JInternalFrame, JLayeredPane, RootPane)
- 7.9 Controles Básicos (JButton, JCheckBox, JComboBox, JList, JMenu, JRadioButton, JSlider, JSpinner, JTextField, JPasswordField). Controles Altamente Formateados (JColorChooser, JEditorPane, JTextPane, JFileChooser, JTable, JTextArea, JTree), Controles No Editables (JLabel, JProgressBar, JSeparator, JToolTip)
- 7.10 Pluggable Look & Feel. Internacionalización y Localización.
- 7.11 Integración con Escritorio Multiplataforma. Integración con Barra de Tareas o Area de Notificación
- 7.12 TP VII: Diseño de GUI.
- 8.0 Programación Orientada a Objetos VII Persistencia**
- 8.1 Almacenando objetos en el disco: Serialización. Grabando y leyendo objetos. Consideraciones de seguridad, almacenamiento y recuperación. Ejemplos. Ejercitación.
- 8.2 Acceso a disco y directorios. La Clase `File`.
- 8.3 Implementación de archivos relativos: La Clase `RandomAccessFile`.
- 8.4 Streams. Jerarquía de clases de I/O. Stream de bajo nivel, orientados a bytes y caracter. Stream de alto nivel, orientados a bytes y caracter.
- 8.5 Operaciones tradicionales de I/O. Ejemplos y ejercitación con archivos secuenciales, estructuras de iteración, combinación con archivos relativos, corte de control.
- 8.6 Más allá de los archivos: los sistemas de bases de datos. Introducción. Breve definiciones de: SBD, CBD, BD, SGBD. Independencia lógica y física. Separación entre programas y datos.
- 8.7 El framework JDBC. Componentes JDBC: API, Driver Manager, Test Suite, ODBC Bridge.
- 8.8 Modelos de Procesamiento Two-Tier y Three-Tier.
- 8.9 Utilizando JDBC: Instalando RDBMS, Drivers, Estableciendo una conexión, Ejecutando consultas, Recuperando filas, Actualizando filas.
- 8.10 Usando Prepared Statements, Transacciones (insert, upate, delete), Ejecutando Store Procedures.
- 8.11 Paquetes requeridos, capturando excepciones, warnings.
- 8.12 Ejemplos y ejercitación: conectando una base de datos, implementando consultas y transacciones, actualizando GUI.
- 8.13 TP VIII: Altas, Bajas, Modificaciones: GUI + JDBC.

3.0 Bibliografía

Bibliografía Básica

- Eckel Bruce, "Thinking in Java / Piensa en Java", 2da. Ed., Prentice Hall, 2002, ISBN: 84-205-3 192-8
- McLaughlin Brett, Pollice Gary, West David, "Head First - Object-Oriented Analysis & Design", O'Reilly Media Inc., 2006, ISBN-10:0-596-00867-8
- O'Docherty, Mike, "Object-Oriented Analysis and Design", John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, 2005, ISBN-13 978-0-470-09240-8

Bibliografía Adicional

- Ball Jennifer, Carson Debbie, Evans Ian, Fordin Scott, Haase Kim, Jendrock Eric, "The Java™ EE 5 Tutorial", Sun Microsystems, Inc., 2006
- Eckel Bruce, "Thinking in Enterprise Java", 2003
- Goodman, Danny, "Javascript Bible – Gold Edition", Hungry Minds, 2001, ISBN: 0-7645-4718-6
- Hanumant Deshmukh, Jignesh Malavia, Matthew Scarpino, "Java Web Component Developer Certification", Manning Publications Co., 2005, ISBN 1-932394-38-9
- Inderjeet Singh, Beth Stearns, Mark Johnson, and the Enterprise Team, "Designing Enterprise Applications with the J2EE™ Platform, Second Edition", Addison-Wesley, Sun Microsystems, Inc., 2002, ISBN 0-201-78790-3
- Philip Heller, Simon Roberts, "Complete Java® 2 Certification: Study Guide, Fifth Edition", Sybex, San Francisco, London, ISBN: 0-7821-4419-5
- Pressman, "Ingeniería de software: un enfoque práctico", Mc Graw Hill, 1998
- Sanghera Paul, Ph.D "SCJP Exam for J2SE 5", Apress, 2006, ISBN-13 (pbk): 978-1-59059-697-5
- Sean Brydon, Greg Murray, Vijay Ramachandran, Inderjeet Singh, Beth Stearns, Thierry Violleau, "Designing Web Services", Sun Microsystems, Inc., 2002

Apuntes de la Asignatura

- Cherencio, Guillermo, "Trabajando con el Compilador GNU C/C++ GCC", ISDFyT N° 189, asignatura: "Programación I", 2008
- Cherencio, Guillermo, "Programando En Capas Usando Java", CISCO Networking Academy, UTN FRD, 2004
- Wheeler, David A., "Program Library HOWTO", <http://www.dwheeler.com/programlibrary>, 2000. Documento cubierto por GNU General Public License (GPL).

Otras fuentes de Información

- BlueJ: <http://www.bluej.org>
- Java Tutorial: <http://java.sun.com/docs/books/tutorial>
- Java Hispano: <http://www.javahispano.org>

Abril 2021

4.0 Evaluación

Se considerara alumnos regulares en modalidad presencial a aquellos que:

- Aprueben el 1er y 2do parcial con nota mayor igual a 4.
- Todos los parciales tienen una instancia de recuperación
- Tengan aprobados todos los trabajos prácticos propuestos
- Tengan aprobado la primera versión del trabajo práctico final integrador.
- Cumplan con todos los requisitos de promoción indicados en el Plan Institucional de Evaluación, normativas de la DGyCE, DES, Pcia. Bs. As.

Se considerará alumnos regulares en modalidad no-presencial o híbrida y mientras dure el aislamiento social obligatorio a aquellos que:

- Mantengan el ingreso y participación en clases sincrónica y lectura de material asincrónico acorde con los registros del campus institucional <http://campus.isft189.edu.ar/moodle>
- Tengan aprobados todas las actividades prácticas propuestas, a través de campus institucional
- Cumplan con todos los requisitos de promoción indicados en el Plan Institucional de evaluación, normativas de la DGyCE, DES, Pcia. Bs. As.

Evaluación Final para Alumnos libres (inscriptos así a comienzo del año) y Regulares:

- Presentación del Trabajo Práctico Final Individual Integrador (aprobado previamente por el docente en la regularización de la asignatura), software aplicativo funcionando, completo y sobre el cual se solicitarán cambios a realizar. El alumno contará con una 1 hora para implementar los cambios.

Criterio de Evaluación:

- En cuanto a los contenidos teóricos se requerirá la comprensión básica de los mismos y su correcta aplicación.
- Los alumnos tendrán a su disposición la sintaxis básica del lenguaje de programación Java, no siendo este punto de interés en la evaluación
- Es de interés para la evaluación que el alumno realice un diseño orientado a objetos que sea aplicable al problema y al mismo tiempo esté acorde con las reglas de un buen diseño
- Es de interés para la evaluación el progreso que el alumno manifieste en cada etapa. La actitud frente a las dificultades, las mejoras introducidas a la aplicación entre la primera versión y la versión final de la misma.
- Es de interés para la evaluación la actitud del alumno frente al trabajo propuesto, las ganas, el interés demostrado en las actividades.
- Es de interés para la evaluación que el alumno adquiera destreza en la utilización de documentación y manuales técnicos en inglés.

Es de interés para la evaluación que el alumno pueda integrar los conocimientos y el trabajo realizado en las asignaturas Análisis de Sistemas y Base de Datos.

Abril 2021

5.0 Cronograma

Semana	Actividad
1	Contrato Pedagógico. 1.1 – 1.3
2	1.4 – 1.9. Entrega TP I
3	2.1 – 2.3
4	2.4 – 2.6
5	2.7 – 2.10
6	2.11 – 2.14 Entrega TP II
7	3.1 – 3.4
8	3.5 – 3.10 Entrega TP III
9	4.1 - 4.7
10	4.8 – 4.15 Entrega TP IV
11	5.1 – 5.5
12	Primer Parcial (Unidades I, II, III, IV)
13	5.6 – 5.9
15	5.10 – 5.18 Entrega TP V
16	6.1 – 6.6
17	6.7 – 6.14 Entrega TP VI
18	7.1 – 7.4
20	7.5 – 7.8
21	7.9 – 7.11 Entrega TP VII
22	Segundo Parcial (Unidades V, VI, VII)
23	8.1 – 8.5
24	8.6 – 8.13 Entrega TP VIII (final integrador)
25	Recuperatorios, Trabajo Integrador
26	Recuperatorios, Trabajo Integrador, Cierre

Abril 2021



6.0 Observaciones

La primera unidad sirve a modo de repaso, se retoma el lenguaje C del año anterior y se implementan librerías dinámicas cargables en tiempo de ejecución.

Luego el alumno se lo introduce en la programación orientada a objetos, haciendo analogías con los conceptos aprendidos el año anterior. Se destaca la necesidad de utilizar lenguajes de alto nivel para evitar la complejidad propia del computador y la necesidad de utilizar diagramas (en este caso, diagramas UML) para una representación gráfica de clases y relaciones. Se avanza muy rápido en la Unidad IV debido a que el alumno reutiliza todos sus conocimientos del lenguaje C. En la unidad 5 profundizan aspectos propios de la programación orientada a objetos y formas de reuso de código muy distintas a la programación procedural. Se organiza el trabajo en paquetes de clases agrupadas por funcionalidad. Se profundiza en el almacenamiento secundario: archivos y framework JDBC para acceso a bases de datos, en este punto se articula en forma horizontal con la asignatura Base de Datos. Se dedica tiempo al diseño de interfaces gráficas, siendo el TP Final Integrador la implementación de un sistema de tipo cliente – servidor con interfaz gráfica implementada con el framework Swing.

Todo el material de estudio (incluso este documento), más software de desarrollo, tutoriales, trabajos prácticos, programas de demostración, etc. requerido por los alumnos en esta asignatura se encuentran catalogados en <http://www.grch.com.ar/docs/poo>

Abril 2021



7.0 Intervención Didáctica –Recursos

Estrategias Didácticas a utilizar:

Según los objetivos de cada clase, las mismas serán:

- Clase expositiva para la explicación de conceptos teóricos.
- Trabajos de producción individual para ejercicios en clase en donde el alumno deba aplicar conceptos teóricos a casos particulares.
- Trabajos de producción grupal cuando el alumno deba debatir con sus pares las distintas propuestas de solución a una situación planteada.

Recursos a utilizar:

- Centro de Cómputos para realizar prácticas en donde el alumno deba aplicar conceptos teóricos a la resolución de situaciones problemáticas.
- Tiza, Pizarrón
- Página web de la asignatura en donde se encuentra digitalizado el material teórico a utilizar, así como también todo el software GNU requerido por la asignatura.
- Notebook personal

Abril 2021

8.0 Actividades de Extensión

Los alumnos deberán especificar el trabajo final integrador a implementar, aquí se espera que los alumnos puedan vincular esta asignatura con sistemas reales que se implementen en el mercado local. No se pretende que el alumno implemente un sistema completo, pero si se espera que pueda producir una aplicación gráfica de tipo cliente – servidor, utilizando base de datos relacional, que integre todos los conceptos vistos en la asignatura y tenga utilidad real.

Abril 2021



9.0 Integración Curricular

Integración Curricular Vertical

Se utilizan todos los conocimientos obtenidos en Programación I para la implementación de métodos. También la similitud entre C y Java es aprovechada para adquirir rápidamente la sintaxis del lenguaje Java. La asignatura aporta los conocimientos de arquitectura cliente – servidor y programación que son requeridos por la asignatura Práctica Profesional.

Integración Curricular Horizontal

Los contenidos están relacionados horizontalmente con la asignatura Base de Datos, se realiza el código de aplicación de base de datos implementada en dicha asignatura, de esta forma el alumno complementa su trabajo en ambas asignaturas. La unidad 1 tiene conceptos vinculados con Sistemas Operativos. Hay material en idioma inglés que son utilizados y en parte traducidos / interpretados en Inglés II. Esto genera en el alumno la idea de articulación entre las asignaturas y no la ilusión de ser simples estancos de conocimiento que no tienen conexión alguna.

Abril 2021