



Nombre de la materia : **Programación de computadoras II**
Clave: **CI0002-T (FALTA VERIFICAR SI ES LA CORRECTA)**
No. De horas /semana : **4**
Duración semanas: **16**
Total de Horas : **64**
No. De créditos : **8**
Prerrequisitos : **CI0000-T (Programación de computadoras)**

Objetivo

Que el alumno mejore su capacidad de desarrollo de software así como su productividad al producir código fácil de mantener y altamente reutilizable, aprovechando las ventajas del encapsulamiento de datos, la herencia, la abstracción de datos y demás ventajas que ofrece la programación orientada a objetos. Al finalizar, el alumno tendrá los conocimientos necesarios para aprobar el examen SCJP.

Contenido sintético

	Horas
Parte I. Generalidades	
1. Introducción	4
2. Datos y Expresiones	4
3. Uso de Clases y Objetos	6
4. Control de Flujo	4
Examen	2
Parte II. Programación Orientada a Objetos	
5. Modelado de Objetos	8
6. Diseño de Clases en Java	14
Examen	2
Parte II. Programación avanzada	
7. Uso de Collections y Generics	4
8. Concurrencia	8
9. Java I/O	6
Proyecto de aplicación	2
Total	64

Bibliografía básica

Paul Deitel and Hayvey M. Deitel. “Java How to Program”. Ninth edition. Prentice Hall. 2011

Katherine Sierra, Bert Bates. “SCJP Sun Certified Programmer for Java 6 Exam 310-065”, McGraw-Hill Osborne Media; 1a edición (Junio 24, 2008).

Danny Poo & Derek Kiong. Object Oriented Programming and Java. Springer 2008



C. Thomas Wu. Introducción a la programación Orientada a Objetos con Java. Mc Graw Hill. 2001

Lewis & Loftus. “Java Software Solutions: Foundations of Program Design”, Addison-Wesley, 6a Edición.

Metodología de enseñanza-aprendizaje

Revisión de conceptos, análisis y solución de problemas en clase:	(X)
Lectura de material fuera de clase:	(X)
Ejercicios fuera de clase (tareas):	(X)
Investigación documental:	()
Elaboración de reportes técnicos o proyectos:	(X)
Prácticas de laboratorio en una materia asociada:	()
Visitas a la industria:	()

Metodología de evaluación:

Asistencia:	(X)
Tareas:	(X)
Elaboración de reportes técnicos o proyectos:	()
Exámenes de Academia o Departamentales	(X)

Contenido desarrollado

- 1. Introducción.** 4 horas
 - 1.1 Propiedades deseables: Reusabilidad y extensibilidad, entre otras.
 - 1.2. Programación orientada a objetos.
 - 1.3. Ingeniería de Software. Ventajas de la Programación Orientada a Objetos.
 - 1.4. El lenguaje de programación Java.
 - 1.5. Ciclo de desarrollo de programas en Java.

- 2. Datos y Expresiones.** 4 horas
 - 2.1. Tipos de datos en java (primitivos, Clases, Interfaces, Enumeraciones).
 - 2.2. Variables y asignación.
 - 2.3. Cadenas de caracteres.
 - 2.4. Expresiones.
 - 2.5. Conversión de datos.
 - 2.6. Clases wrapper.

- 3. Uso de Clases y Objetos.** 6 horas
 - 3.1. La API de Java.
 - 3.2. Creación de objetos.
 - 3.3. La clase String
 - 3.4. Paquetes
 - 3.5. La clase Random.



3.6. La clase Math.

3.7. Formateo de Salida

4. Control de Flujo

4 horas

4.1. Sentencia if.

4.2. Comparando datos.

4.3. Sentencia switch.

4.4. Sentencia while.

4.5. Sentencias do y for.

4.6. Iteradores.

4.7. Recursividad.

4.8. Assertions.

4.9. Introducción a Excepciones

5. Modelado de Objetos.

8 horas

5.1. Modelos de especificación del Dominio: La técnica CRC.

5.2. Diagrama de Clases (modelo conceptual).

5.3. Ejemplos prácticos para descubrir Clases y Métodos utilizando la Técnica CRC.

5.4. Utilizando la Técnica CRC, encontrar responsabilidades para las Clases y Colaboraciones.

5.5. Introducción a especificación de modelos con el Lenguaje de Modelado Unificado (UML). Tipos de Diagramas UML Diagramas de clases. Elementos del Diagrama de Clases. Relaciones entre clases: Asociación, Composición, Agregación, Dependencia, Herencia, Ejemplos de relaciones entre clases

6. Diseño de Clases en Java

14 horas

6.1. Repaso de Clases y Objetos.

6.2. Anatomía de una Clase (java y UML).

6.3. Encapsulación.

6.4. Anatomía de un método.

6.5. Revisión de constructores.

6.6. Miembros estáticos.

6.7. Relaciones entre clases.

6.8. Interfaces.

6.9. Herencia.

6.10. Subclases.

6.11. Sobreescritura de métodos.

6.12. Visibilidad.

6.13. Polimorfismo.

6.14. Polimorfismo usando herencia.

6.15. Polimorfismo usando Interfaces.

6.16. Revisión de Excepciones.

7. Uso de Collections y Generics.

4 horas

7.1. Uso de == y del método equal, y sobreescritura del método hashCode.

7.2. Clases de la API Collections (Set, List, Map, etc).

7.3. Generics.



7.4. Uso de `java.util.Comparator` and `java.lang.Comparable` así como el paquete `java.util` para escribir código que ordene y realice búsquedas.

8. Concurrencia 8 horas

8.1. Creación de hilos a partir de `java.lang.Thread` y `java.lang.Runnable`.

8.2. Estados de un hilo.

8.3. Solución de problemas de acceso concurrente.

8.4. Uso de `wait`, `notify`, or `notifyAll`.

9. Java I/O. 2 horas

9.1. Uso de `BufferedReader`, `BufferedWriter`, `File`, `FileReader`, `FileWriter` and `PrintWriter`.

9.2. Serialización de objetos usando las siguientes clases: `DataOutputStream`, `FileInputStream`, `FileOutputStream`, `ObjectInputStream`, `ObjectOutputStream` and `Serializable`.

9.3. Uso de la clase `java.util.Locale`.

9.4. Uso de expresiones regulares.

Historia de revisiones

Programa propuesto por los Dres. José Antonio Camarena Ibarrola y Carlos Lara para el semestre Agosto 2009/ Enero de 2010.

Programa modificado por Dr. Félix Calderón Solorio. Mayo de 2017.

Aprobación por el H. Consejo Técnico de la FIE: 10 de abril de 2018