

Código: DI-DUSAR-I-07

Fecha: 01-02-2017

Versión: N° 5

**SÍLABO 2023-1**

FACULTAD DE INGENIERÍA Y ARQUITECTURA

CARRERA DE INGENIERÍA DE SISTEMAS

|  |  |
| --- | --- |
| ASIGNATURA | **PROGRAMACIÓN ORIENTADA A OBJETOS** |
| ÁREA | INGENIERÍA DE SOFTWARE |
| CÓDIGO | 650005 |
| NIVEL | CUARTO |
| CARÁCTER | OBLIGATORIO |
| REQUISITOS | INTRODUCCIÓN A LA PROGRAMACIÓN ESTRUCTURAS DISCRETAS DE COMPUTACIÓN |
| CRÉDITOS | TRES (3) |
| HORAS DE TEORÍA SEMANAL | UNO (1) |
| HORAS DE PRÁCTICA SEMANAL  IDIOMA  PROFESOR(ES) | CUATRO (4)  ESPAÑOL  HERNAN NINA HANCO (Coordinador)  PABLO ALBERTO ROJAS JAEN  JUAN CARLOS EDUARDO ROMAINA ACEVEDO |

1. **SUMILLA**

La asignatura trata sobre el paradigma de programación orientada a objetos. El estudiante utilizará los conceptos de programación orientada a objetos en la implementación de algoritmos y estructuras de datos para desarrollar soluciones a problemas de diversos rubros. Los temas tratados son los siguientes: conceptos y principios básicos de la programación orientada a objetos (POO), asociaciones entre clases, herencia y clases abstractas, polimorfismo, introducción a patrones e interfaces gráficas

1. **OBJETIVO GENERAL**

Al finalizar la asignatura, el alumno será capaz de interpretar y aplicar los conceptos fundamentales comprendidos en el paradigma de la programación orientada a objetos; para de esta forma, plantear y desarrollar aplicaciones que den solución a problemas del mundo real utilizando técnicas, modelos y estructuras propias de este paradigma.

1. **OBJETIVOS ESPECÍFICOS**
   1. Conocer las diferencias fundamentales entre clases, objetos y métodos.
   2. Modelar sus soluciones haciendo uso del diagramado de clases en UML.
   3. Aplicar los conceptos de herencia y clases derivadas.
   4. Desarrollar aplicaciones gráficas GUI.
   5. Introducir al alumno hacia el manejo de Excepciones, patrones y conectividad a Base de Datos
2. **PROGRAMA ANALÍTICO**

**UNIDAD 1: INTRODUCCIÓN A LA PROGRAMACIÓN ORIENTADA A OBJETOS**

**05 horas**

**PRIMERA SEMANA**

INTRODUCCIÓN AL PARADIGMA DE ORIENTACIÓN A OBJETOS.

Evolución de la programación desde la programación estructurada a la programación orientada a objetos. Flujo de datos: secuencia, selección, repetición y modularidad.

**UNIDAD 2: CLASES Y OBJETOS**

**15 horas**

**SEGUNDA SEMANA**

OBJETOS – CLASES

Estructura de un objeto. Encapsulado, atributos, atributos del tipo arreglo, estructura básica de una clase. Instanciación de objetos. Visibilidad de clase.

**TERCERA SEMANA**

OBJETOS – CLASES

Constructores. Métodos accesadores y modificadores. Métodos de instancia y métodos de clase.

**CUARTA SEMANA**

CLASES – OBJETOS – MÉTODOS.

Sobrecarga de métodos.

**UNIDAD 2: HERENCIA Y POLIMORFISMO**

**15 horas**

**QUINTA SEMANA**

HERENCIA – ATRIBUTOS - MÉTODOS

Definición de Herencia Simple. Clases y Super Clases. Métodos redefinidos. Construcción de una Jerarquía de clases. Representación de la herencia con UML.

**SEXTA SEMANA**

POLIMORFISMO

Definición de polimorfismo. Compatibilidad de tipos. Métodos polimórficos. Métodos estáticos y dinámicos.

**SÉPTIMA SEMANA**

ARREGLO DE OBJETOS.

Arreglos de primitivos. Arreglo de referencias de objetos. Recorridos, ingreso y validaciones de objetos a arreglos.

**OCTAVA SEMANA 05 horas**

Cierre obligatorio de evaluaciones: aplicación de la última evaluación de la EC1.

Repaso de temas o revisión de evaluación.

**UNIDAD 3: CLASES ABSTRACTAS E INTERFASES**

**20 horas**

**NOVENA SEMANA**

CLASES ABSTRACTAS.

Definición de una clase abstracta. Declaración de clases abstractas y métodos abstractos. Implementación de métodos abstractos. Colecciones heterogéneas.

**DECIMA SEMANA**

DISEÑO DE INTERFACES.

Definición, reglas y uso de interfaces. Diferencias entre clases abstractas y clases interfaces.Herencia múltiple. Desarrollo de ejercicios usando interfaces.

**DÉCIMO PRIMERA SEMANA**

ASOCIACIÓN ENTRE CLASES

Implementación de asociación entre clases. Cardinalidades y restricciones. Agregación y Composición. Clases anidadas. Representación de asociación de clases en UML. Patrones de Diseño Orientado a Objetos.

**DÉCIMO SEGUNDA SEMANA**

MENSAJES ENTRE CLASES

Invocación de métodos. Paso de Mensajes entre clases. Variables de referencia.

**UNIDAD 4: INSTERFAZ GRÁFICA DE USUARIO**

**5 horas**

**DÉCIMO TERCERA SEMANA**

INTERFACES GRAFICAS GUI

Definición de interfaces gráficas. Manejo de eventos. Desarrollo de aplicaciones simples utilizando componentes gráficos.

**UNIDAD 4: CLASES GENÉRICAS**

**10 horas**

**DÉCIMO CUARTA SEMANA**

CLASES GENÉRICAS

Definición y uso de clases genéricas (templates). Convención para definición de parámetros.

**DÉCIMO QUINTA SEMANA**

CLASES GENÉRICAS – VARIANTES

Extensión de Clases genéricas. Clases genéricas comparables. Estructuras de Datos auxiliares: LinkedList

**DÉCIMO SEXTA SEMANA 05 horas**

Cierre obligatorio de evaluaciones: aplicación de la última evaluación de la EC2.

Repaso de temas o revisión de evaluación.

**DÉCIMOSÉPTIMA SEMANA**

Entrega de notas.

1. **METODOLOGÍA DE ENSEÑANZA - APRENDIZAJE**

La Universidad de Lima ha diseñado la **Metodología IATC** para orientar el proceso de enseñanza-aprendizaje y asegurar el logro de los objetivos de la siguiente manera:

* Impacto: motivar y generar curiosidad. Presentar objetivos y agenda.
* Adquisición del aprendizaje: transmitir el conocimiento con estrategias innovadoras. Promover la interacción.
* Transferencia de lo aprendido: desarrollar actividades significativas. Utilizar estrategias y técnicas didácticas.
* Cierre del aprendizaje: concluir sobre el aprendizaje. Reflexionar sobre el logro del objetivo.

Además, la asignatura se desarrollará tomando en cuenta las siguientes metodologías:

1. Revisión previa de los temas de clase, por parte de los alumnos, bajo el concepto de aula invertida.
2. Desarrollo participativo de las sesiones y temas del curso, con enfoque teórico-práctico, propiciándose que los alumnos desarrollen ejercicios individuales y grupales en clase.
3. Autoevaluación de los alumnos, mediante la asignación de ejercicios selectos para resolución fuera de aula y posterior validación contra la solución del grupo.
4. Consolidación de los conceptos, mediante la presentación de casos reales de aplicación, generando discusión sobre otros escenarios donde podrían aplicarse.

Adicionalmente, los profesores brindarán asesorías a los alumnos según el rol establecido, para atender consultas y resolver dudas sobre la asignatura.

1. **SISTEMA DE EVALUACIÓN**

Para el sistema de evaluación, esta asignatura es de tipo:

|  |
| --- |
| **Teórico práctico** |

La nota final de la asignatura (NF) es el promedio ponderado de las notas obtenidas en las siguientes evaluaciones:

|  |  |
| --- | --- |
| Evaluación Continua 1 (EC1) | 50 % |
| Evaluación Continua 2 (EC2) | 50 % |

La nota de EC1 y de EC2 comprende:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EC** | **Semana** | **Tipo de evaluación** | **Peso** | **Objetivo /**  **Competencia a evaluar** |
| **EC1**  **(50 %)** | 4 | Examen Escrito (EE) | 50% | Objetivos: 1 y 2 |
| 8 | Examen Escrito (EE) | 50% | Objetivos: 3 |
| **EC2 (50%)** | 14 | Trabajo de investigación (TI) | 60% | Objetivos: 1 y 3 |
| 16 | Examen Escrito (EE) | 40% | Objetivos: 4 y 5 |

**VII. BIBLIOGRAFÍA**

**OBLIGATORIA**

1. Sarcar, V. (2016). *Interactive Object-Oriented Programming in Java*. Apress, Berkeley, CA.
2. Juneau J. (2017). *Java 9 Recipes A Problem-Solution Approach*. Apress, Berkeley, CA.

**COMPLEMENTARIA**

1. Sánchez Allende, Jesús. Programación en Java. McGraw-Hill, 2009 Código Biblioteca U. Lima: 005.133J P
2. Sun Microsystems, Inc. *Java Programming Language SL-275-SE6*. Código Biblioteca U. Lima: 005.133J J T.1.
3. Weitzenfeld, A. (2004). Ingeniería de software orientada a objetos con UML, Java e Internet. Editorial Thomson.
4. Phillips, D. (2015). Python 3 object-oriented programming. Packt Publishing Ltd.
5. Robert Lafore (2002). Object-Oriented Programming in C++, Fourth Edition. Sams Publishing