|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAMA UNIDAD CURRICULAR** | | | | | | | | |
| **Unidad Académica** | | | DEPARTAMENTO DE ECONOMÍA, PRODUCCIÓN E INNOVACIÓN TECNOLÓGICA | | | | | |
| **Carrera/s** | | | LICENCIATURA EN GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN | | | | | |
| **Plan de Estudios** | | | Resolución (CS) 220/2019 | | | | | |
| 1. **Datos sobre la unidad curricular** | | | | | | | | |
| **Nombre** | **Programación Orientada a Objetos** | | | | | **Código** | 6018 | |
| **Modalidad** | Presencial | | **Régimen** | | Cuatrimestral | | | |
|
| **Equipo responsable** | | | Ciarallo, Cristian  Blackman, Gastón | | | | | |
| **Año y mes de presentación del programa** | | | 2023-09 | | | | | |
| 1. **Carga horaria** | | | | | | | | |
| **Horas de clase semanales** | | 4 | |  | | | |  |
| **Horas de clase totales** | | 64 | | Horas totales teóricas | | | |  |
| Horas totales prácticas | | | |  |
| Otras horas totales (laboratorio, trabajo de campo, etc) | | | |  |

|  |  |
| --- | --- |
| 1. **Unidades correlativas** precedentes en el Plan de Estudios | |
| Denominación | Código |
| Algoritmos y Programación | 6008 |

|  |
| --- |
| 1. **Contenidos mínimos** según Plan de Estudios |
| Problemas de las técnicas procedurales. Clases y Objetos. Programas orientado a objetos. Encapsulamiento. Atributos. Propiedades de los atributos. Mensaje y método. Jerarquías de clases. Herencia. Clases abstractas. Polimorfismo y binding dinámico. Diseño de objetos complejos. Relaciones entre objetos. Objetos contenedores. Colecciones de objetos. Lenguajes orientados a objetos. |

|  |
| --- |
| 1. **Fundamentación** |
| La Programación Orientada a Objetos tiene como propósito la solución de problemas mediante el desarrollo de programas de computadoras, identificando a la misma dentro del paradigma de objetos. Esta asignatura resulta de gran importancia en la formación de los futuros Analistas Programadores, permitiéndoles un desarrollo profesional y brindándoles herramientas de tipo cognitiva que permiten emprender la búsqueda de la solución de una manera natural, viendo la interacción de los objetos en el mundo real con facilidad y claridad.  Contribuye al razonamiento lógico de los estudiantes desarrollando habilidades intelectuales que le permitirán clasificar, describir y ordenar el comportamiento y la comunicación de los objetos con el fin de desarrollar programas de computadoras que presenten soluciones claras, eficaces y eficientes a las problemáticas informáticas cotidianas.  Esta asignatura ayuda a la concepción y modelo de la realidad de manera objetiva facilitando así el establecimiento de relaciones multidisciplinarias, en cuanto al contenido de los problemas que se resuelven, potenciando las habilidades de modelar, diseñar y desarrollar software. |

|  |
| --- |
| 1. **Objetivos** |
| El objetivo general de esta asignatura es que los alumnos desarrollen capacidades analíticas y prácticas para la comprensión de los conceptos fundamentales inherentes al proceso de desarrollo de software. Entender a la programación orientada a objetos como herramienta fundamental del desarrollo profesional e inserción laboral.  El objetivo especifico de POO es **catalogar y diferenciar el código, en base a estructuras jerárquicas dependientes, al estilo de un árbol genealógico**. Los objetos se crean a partir de una serie de especificaciones o normas que definen como va a ser el objeto, esto es lo que en POO se conoce como una clase. |

|  |
| --- |
| 1. **Contenidos (**organizados por unidades) |
| **Unidad I: Repaso Tipo Abstractos de Datos. Clases y Objetos. Jerarquías de clases.**  Modularidad, principios de abstracción en lenguajes de programación. Abstracción de datos y abstracción procedimental, Especificación de los TAD. Modelos de Datos Genéricos, Generics en Java. Encapsulamiento Clases y objetos. Jerarquía de clases: Relación isA/hasA. Generalización / Especialización. Herencia, Herencia simple, Herencia múltiple, Interfaz y clases abstractas. Genericidad, Polimorfismo y binding dinámico.  **Unidad II: Diseño de Objetos. Colecciones de Objetos e iteradores.**  Diseño de objetos complejos. Relaciones entre Objetos. Relación de conocimiento. Relación isPartOf.  Características y utilización de colecciones, bucles, iteradores y Arrays: interfaz Collection, clases ArrayList, LinkedList, HashSet, TreeSet, interfaz Map, clases TreeMap, HashMap, Iteradores: interfaz Iterator  **Unidad III: El lenguaje de Modelado Unificado (UML). Aplicación de técnicas orientadas a objetos para la construcción de aplicaciones con interfaces visuales, patrón MVC**  Introducción a UML. Manejo de diagramas y características comunes. Diagrama de clases y de objetos, Diagramas de caso de uso, Diagrama de secuencia.  Mecanismos de Dependencias. Diseño de Interfaces: Model-View-Controller. Métodos changed: y update: Introducción a la persistencia de objetos: serialización. |

|  |
| --- |
| 1. **Bibliografía obligatoria y complementaria (organizada por unidades)** |
| **Unidad I:**  [Fontela Carlos M.] Programación Orientada a Objetos 2da. Edición Java y UML. Año Edición 2011. Editorial Nueva Liberia.  [Luis Joyanes Aguilar- Ignacio Zahonero Martinez] Estructura de datos en Java. 1ra Edición. Ed. Mc Graw Hill, 2008.  **Unidad II:**  [Fontela Carlos M.] Programación Orientada a Objetos 2da. Edición Java y UML. Año Edición 2011. Editorial Nueva Liberia.  [Luis Joyanes Aguilar- Ignacio Zahonero Martinez] Estructura de datos en Java. 1ra Edición. Ed. Mc Graw Hill, 2008.  **Unidad III:** [Fontela Carlos M.] Programación Orientada a Objetos 2da. Edición Java y UML. Año Edición 2011. Editorial Nueva Liberia.  [Luis Joyanes Aguilar- Ignacio Zahonero Martinez] Estructura de datos en Java. 1ra Edición. Ed. Mc Graw Hill, 2008.  [Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides] Patrones de Diseños, Elementos del software orientado a elementos reutilizables. Pearson educación 2003. |

|  |
| --- |
| 1. **Metodología de trabajo** |
| En las clases teóricas se desarrollarán los contenidos de cada unidad mediante una exposición oral de los docentes, promoviendo la participación de los alumnos con diversos ejemplos de aplicación.  En las clases prácticas,se aplicarán los conceptos impartidos en las clases teóricas reforzando los mismos y fomentando el extreme programming para resolver los problemas prácticos, discutiendo estrategias de análisis y diseño.  El desarrollo de los trabajos prácticos implica el diseño de programas, su implementación en lenguaje JAVA y su prueba a través de testeos.  **Los recursos metodológicos que se utilizan en cada Unidad Didáctica son los siguientes:**   * Clases expositivas explicativas. * Participación de los estudiantes en actividades no prolongadas en clase de fijación de conceptos teóricos expuesto. * Aplicación, discusión y resolución de problemas. * Desarrollo de software en grupo mediante extreme programming.   **Los trabajos prácticos son los siguientes:**   * TP0: Repaso del concepto de TDA. * TP1: Modelado de una solución al requisito de sistemas informáticos utilizando el paradigma de POO y las herramientas UML: Diagramas de clases y secuencia. Implementación en código JAVA. * TP2: Identificación de casos de uso a partir de requerimientos planteados en requisitos de sistemas informáticos. * TP3: Trabajo practico final integrador, implementación de la solución completa a un sistema informático requerido de tamaño pequeño. |
|  |

|  |
| --- |
| 1. **Evaluación** |
| 2 evaluaciones parciales presenciales y 1 TP integrador.  En todas las evaluaciones, se computará como insuficiente cualquier cifra con fracciones que no alcance 4 (cuatro) puntos.  Los estudiantes que mantengan condición regular podrán aprobarla mediante:   1. Promoción directa (Calificacion de 7 (siete) o más como promedio de todas las instancias evaluativas. 2. Aprobación de examen integrador. 3. Mediante examen final.   Se contará con una instancia de recuperatorio para cada examen parcial y TP, de acuerdo al Régimen General de Estudios de Grado y Posgrado. |

|  |
| --- |
| 1. **Instancias de práctica** (si corresponde) |
|  |

|  |  |
| --- | --- |
| 1. **Cronograma de actividades** | |
| Semana 1 | **Teórica:** Presentación de la materia. Evaluación diagnóstico. Tipo Abstracto de Datos, ocultación – encapsulamiento. Construcción de un TAD en JAVA. Ejemplo de Implementación.  **Práctica:** TP (TAD), Listas, Pilas y Colas. |
| Semana 2 | **Teórica:** Clases y objetos. Instanciación. Comunicación entre objetos – Mensajes y Métodos.  **Práctica**: Continuación de TP. |
| Semana 3 | **Teórica:** Herencia, clases abstractas e interfaz. Polimorfismo sobrecarga.  **Práctica**: Continuación de TP. |
| Semana 4 | **Teórica:** Herencia, clases abstractas e interfaz. Polimorfismo sobrecarga.  **Práctica**: Continuación de TP. |
| Semana 5 | **Teórica:** Características y utilización de colecciones, bucles, iteradores y Arrays: interfaz Collection  El lenguaje de Modelado Unificado (UML).Diagramas de Estructura Estática: Diagramas de Clases Diagramas de Secuencia  **Práctica:** TP - UML Diagramas de Clases y Secuencia. |
| Semana 6 | **Teórica:** clases ArrayList, LinkedList, HashSet, TreeSet, interfaz Map, clases TreeMap, HashMap, Iteradores: interfaz Iterator  **Práctica:** TP – Ejercicios con Iteradores, Arrays y Colecciones |
| Semana 7 | **Teórico:** Diagramas de Estado, Actividad. Ejemplos de aplicación.  **Práctica:** Continuación TP - UML Diagramas de Clases y Secuencia. |
| Semana 8 | **Teórico:** Ejemplos de aplicación UML.  1er Parcial. Teoría y práctica de las clases anteriores hasta lo visto en la semana 6. |
| Semana 9 | **Teórica:** Diagramas de Casos de Uso  **Práctica:** TP. Casos de uso. |
| Semana 10 | **Teórica:** Aplicación de técnicas orientadas a objetos para la construcción de aplicaciones. Separación de persistencia, modelo y presentación.  **Práctica:** Continuación del TP |
| Semana 11 | **Teórica:** Patrón de Arquitectura: Model-View-Controller.  **Práctica:** TP: Implementación de aplicación MVC |
| Semana 12 | **Teórica:** Patrón de Arquitectura: Model-View-Controller.  **Práctica:** Continuación del TP |
| Semana 13 | **Teórica:**. Patrón de Arquitectura: 3 Capas  2do Parcial: Teoría y práctica de las clases anteriores hasta la clase de la semana 11 inclusive. |
| Semana 14 | **Teórica:** Diseño de Interfaces - Interfaz SWING.  Recuperatorio 1er parcial y/o 2do parcial |
| Semana 15 | **Teórica:** Diseño de Interfaces - Interfaz SWING.  **Práctica** – Entrega de notas – |
| Semana 16 | Cierre de la materia |

Firma del docente/s responsable/s:

CRISTIAN CIARALLO

