# PROGRAMA ANALÍTICO

## *1. DATOS INFORMATIVOS*

| | | | |
|---|---|---|---|
| **DEPARTAMENTO:** CIENCIAS DE LA COMPUTACION | | **ÁREA DE CONOCIMIENTO:** PROGRAMACION | |
| **NOMBRE DE LA ASIGNATURA:** PROG. ORIENTADA A OBJETOS | | **PERIODO ACADÉMICO:** PREGRADO S-II NOV20 - ABR21 | |
| **CÓDIGO:** A0J07 | | **No. CREDITOS:** | **NIVEL:** PREGRADO |
| **FECHA ELABORACIÓN:** 09/12/2020 | **EJE DE FORMACIÓN** | **HORAS / SEMANA** | |
| | BÁSICA | **TEÓRICAS:** | **PRÁCTICAS/LABORATORIO** |

**DESCRIPCIÓN DE LA ASIGNATURA:**

La materia Programación Orientada a Objetos, es una asignatura del eje de formación profesional, que se caracteriza por contribuir a la formación de los elementos de competencia y fortalecer las unidades de competencia en análisis, diseño, y construcción de aplicaciones de software, basado en el paradigma
orientado a objetos, sus fundamentos y principios, como el encapsulamiento, la abstracción, la herencia, el polimorfismo apoyados, por el lenguaje de programación Java. Esta asignatura se enfoca principalmente en la resolución de problemas complejos del mundo real, y en producir aplicaciones de calidad, empleando principios y
prácticas de la Ingeniería de Software, tales como pruebas de unidad, patrones de diseño y los “SOLID Principales”. Se
fortalece también con el uso de interfaces gráficas de usuario, y conexión a bases de datos que permiten la adecuada
interacción entre el usuario y el computador.

**CONTRIBUCIÓN DE LA ASIGNATURA A LA FORMACIÓN PROFESIONAL:**

La asignatura contribuye al resultado de aprendizaje del nivel y es parte sustancial de la formación profesional, los componentes son la solución a problemas orientados a la integración de diferentes aplicaciones e infraestructura tecnológica existente en las organizaciones, bajo el sustento de la programación de computadores.

**RESULTADO DE APRENDIZAJE DE LA CARRERA (UNIDAD DE COMPETENCIA):**

Aplica el paradigma de programación orientado a objetos para implementar algoritmos en lenguajes de programación que solucionan problemas básicos en diferentes dominios.

**OBJETIVO DE LA ASIGNATURA:**

**RESULTADO DE APRENDIZAJE DE LA ASIGNATURA: (ELEMENTO DE COMPETENCIA):**

Conceptuales:
Conoce los conceptos de la programación Orientada a Objetos. Describe la relación entre la estructura
estática de la clase y la estructura dinámica de la instancia de la clase.
Procedimentales:
Diseña, implementa, prueba y depura programas sencillos en un lenguaje de programación orientado a objetos. Diseña, implementa y prueba la aplicación relacionada entre objetos utilizando una clase mediante la jerarquía y herencia. Utiliza iteraciones para acceder a los elementos de un contenedor.
Actitudinales:
Participa activamente en un equipo de trabajo desarrollando aplicaciones que empleen conocimientos de programación orientada a objetos

## *2. SISTEMA DE CONTENIDOS Y RESULTADOS DEL APRENDIZAJE*

| **UNIDADES DE CONTENIDOS** | |
|---|---|
| **Unidad 1**<br>PRINCIPIOS DE DISEÑO ORIENTADO A OBJETOS. | **Resultados de Aprendizaje de la Unidad 1**<br>Aplica, utiliza técnicas de programación orientada a objetos (POO), emplea UML para el modelamiento de Clases (Diagramas de clases) y de requerimientos (diagramas de casos de uso), utilizando una herramienta de modelado y un lenguaje POO. |
| **1.1 Sistemas de control de versionamiento (VCS**<br>Software VCS: Git, GitHub.<br>**1.2 Paradigmas de programación**<br>Transición de paradigma<br>**1.3 Entorno de Desarrollo** | |

**UNIDADES DE CONTENIDOS**

Características e Instalación.

Administración y configuración del área de trabajo.

Líneas de Comando

**1.4 Revisión de conceptos generales de la POO.**

Principios Generales de la Programación Orientada a Objetos.

Definición de clases, objetos, atributos y métodos.

**1.5 Modelamiento de clases y Objetos**

UML: Diagramas de Casos de Uso

UML: Diagramas de Clases

Identificación de clases de un sistema, uso correcto de identificadores.

Modificadores de Acceso

Implementación de clases

**1.6 Código limpio**

1.6.1 Estándares de implementación, buenas prácticas de programación.

1.6.2 Atributos de calidad de código

**1.7 Estructura general de un programa**

1.7.1 Creación de un programa básico O.O

1.7.2 Tipos de datos, primitivos y referenciados.

**1.8 Lectura escritura de datos por consola**

1.8.1 Entrada de datos

1.8.2 Salida de datos

**1.9 Excepciones**

Definición

Excepciones y errores

Clases de excepción

Tipos de excepciones.

Excepciones personalizadas.

**1.10 Encapsulamiento**

Definición

Clases

Paquetes

Librerías/Bibliotecas, métodos static

**1.11 Constructores**

Tipos de constructores

Instanciación

**1.12 Métodos getters, setters.**

Definición e Implementación

**1.13 PERSISTENCIA DE DATOS**

MANIPULACION DE ARCHIVOS

LECTURA Y ESCRITURA DE DATOS

LECTURA Y ESCRITURA DE OBJETOS

FORMATOS DE DATOS : CSV, JSON

COLECCIONES / ARRAYLIST

**1.14 ARREGLOS Y COLECCIONES**

ARREGLOS DE DATOS PRIMITIVOS

ARREGLOS DE OBJETOS

# PROGRAMA ANALÍTICO

**UNIDADES DE CONTENIDOS**

**1.15 RELACIONES ENTRE CLASES**

ASOCIACIÓN AGREGACIÓN/COMPOSICIÓN : MODELADO E IMPLEMENTACIÓN

DEPENDENCIA: MODELADO

| **Unidad 2** | **Resultados de Aprendizaje de la Unidad 2** |
|---|---|
| RELACIONES ENTRE CLASES E INTERFACES GRÁFICAS DE USUARIO | Diseña aplicaciones enfocadas a la ingeniería con técnicas de programación orientada a objetos con el uso de interfaz gráfica amigable para el usuario y persistencia de datos. Realiza pruebas de unidad y depuraciones de la aplicación. |

**2.1. Generalización/Especialización**

Herencia: Definición, ventajas, nomenclatura, reglas y modelado.

Implementación.

**2.2. Revisiones de Código**

2.2.1 Revisiones de Código

**2.3. Gestión de Defectos (testing).**

2.3.1. Verificación y Validación

2.3.2. Pruebas Vs Depuración.

2.3.3. Pruebas de unidad

**2.4. Polimorfismo**

2.4.1. Definición y ventajas

2.4.2. Sobrecarga de métodos

2.4.3. Sobre escritura de métodos

2.4.4. Asignación de objetos a variables de su superclase.

**2.5. Interfaces de programación y clases Abstractas**

2.5.1. Definición

2.5.2. Modelado

2.5.3. Declaración e implementación

2.5.4. Clases Abstractas

2.5.5. Métodos Abstractos

2.5.6. Interfaces y polimorfismo.

**2.6. Modelo Vista Controlador**

2.6.1. Arquitectura

2.6.2. Implementación

**2.7. Bases de Datos no SQL**

2.7.1. Acceso a base de datos

2.7.2. Drivers y Conexión

2.7.3. Operaciones CRUD

**2.8. Componentes y objetos gráficos**

2.8.1. Widgets (componentes gráficos)

2.8.2. Formularios

2.8.3. Menús, tablas

2.8.4. Gestión de eventos

2.8.5. Integración de componentes gráficos y clases

| **Unidad 3** | **Resultados de Aprendizaje de la Unidad 3** |
|---|---|
| TÉCNICAS AVANZADAS DE P.O.O. | Diseña e Implementa programas de computación con calidad utilizando la POO, interfaces gráficas, patrones de diseño y acceso a bases de datos. Aplica principios de POO (SOLID). |

**3.1 SOLID Principles**

# PROGRAMA ANALÍTICO

**UNIDADES DE CONTENIDOS**

| | |
|---|---|
| 3.1.1 | Single Responsibility |
| 3.1.2 | Open/Closed |
| 3.1.3 | Liskov Substitution |
| 3.1.4 | Interface Segregation |
| 3.1.5 | Dependency Inversion |
| 3.1.6 | Código entendible, flexible y mantenible |
| **3.2** | **Modularidad** |
| 3.2.1 | Localización de decisiones de diseño |
| 3.2.2 | Alta cohesión |
| 3.2.3 | Bajo acoplamiento |
| **3.3** | **Introducción a Patrones de Diseño** |
| 3.3.1 | Conceptos Generales |
| 3.3.2 | Importancia de los patrones de diseño |
| 3.3.3 | Tipos de patrones |
| **3.4** | **Patrones de creación** |
| 3.4.1 | Singleton |
| **3.5** | **Patrones de estructura** |
| 3.5.1 | Template |
| **3.6** | **Patrones de comportamiento** |
| 3.6.1 | Strategy |

## *3. PROYECCIÓN METODOLÓGICA Y ORGANIZATIVA PARA EL DESARROLLO DE LA ASIGNATURA*

**(PROYECCIÓN DE LOS MÉTODOS DE ENZEÑANZA - APRENDIZAJE QUE SE UTILIZARÁN)**

1. Clase Magistral
2. Resolución de Problemas
3. Prácticas de Laboratorío

## PROYECCIÓN DEL EMPLEO DE LA TIC EN LOS PROCESOS DE APRENDIZAJE

1. Herramientas Colaborativas (Google, drive, onedrives, otros)
2. Material Multimedia
3. Video Conferencia
4. Software de Simulación
5. Aula Virtual

## *4. TÉCNICAS Y PONDERACIÓN DE LA EVALUACIÓN*

- En este espacio se expresarán las técnicas utilizadas en la evaluación del proceso de enseñanza aprendizaje o evaluación formativa y sumativa.
- Las técnicas que se recomienda usar son: Resolución de ejercicios, Investigación Bibliográfica, Lecciones oral/escrita, Pruebas orales/escrita, Laboratorios, Talleres, Solución de problemas, Prácticas, Exposición, Trabajo colaborativo, Examen parcial, Otras formas de evaluación.
- Recordar que mientras más técnicas utilicen, la evaluación será más objetiva y el desempeño del estudiante se reflejará en su rendimiento (4 o 5 técnicas).
- Para evaluar se deberá aplicar la rúbrica en cada una de las técnicas de evaluación empleadas. Se debe expresar en puntaje de la nota final sobre 20 puntos. No debe existir una diferencia mayor a dos puntos entre cada técnica de evaluación empleada.
- En la modalidad presencial existen tres parciales en la modalidad a distancia existen dos parciales, toda la planificación de periodo académico se la realiza en función del número de parciales de cada modalidad.
- La ponderación a utilizarse en la evaluación del aprendizaje del estudiante será la misma en las tres parciales.
- Para la aprobación de una asignatura se debe tener una nota final promedio de 14/20, en los tres o dos

# PROGRAMA ANALÍTICO

## *5. BIBLIOGRAFÍA BÁSICA/ TEXTO GUÍA DE LA ASIGNATURA*

| Titulo | Autor | Edición | Año | Idioma | Editorial |
|---|---|---|---|---|---|
| Introducción a la programación orientada a objetos | Deitel, Paul J | - | 2010 | spa | México : Pearson |
| Introducción a la programación orientada a objetos con java | Wu, C. Thomas | - | 2001 | spa | Madrid : McGraw Hill |
| Introducción a la programación orientada a objetos | Budd, Timothy | - | 1994 | Español | : Addison-Wesley Iberoamericana |