	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

## GUÍA DE LABORATORIO DE PROGRAMACIÓN ORIENTADA A OBJETOS

### Unidad didáctica 1: Diseño Orientado a Objetos

#### Eje Temático: Diseño e implementación de clases

No. Guía		2		<b>Resultados de Aprendizaje de la Unidad Didáctica</b>  Escribir correctamente una clase, incluyendo sus atributos y métodos básicos con el fin de apropiar la sintaxis de lenguaje  Realizar consultas bibliográficas de diferentes fuentes Comprender los fundamentos de Programación Orientada a Objetos.  Consultar sobre un lenguaje orientado a objetos diferente a los tradicionales.  Elaborar un ejemplo de uso del lenguaje en cuanto a clases y objeto	
2 sesiones		3 y 4 Semana			
Horas de Trabajo					
Trabajo con Docente		Trabajo Autónomo			
6		12			
Tipo de trabajo					
Grupal	X	Ind		Laboratorio Requerido	Asistido por computador / Laboratorio de Informática

### Introducción

Esta guía se realiza durante 4 sesiones, se propone la siguiente distribución de actividades para realizar el trabajo de laboratorio, sin embargo, avanza a tu ritmo:

#### Sesión 1:

1. Resuelva las preguntas de la sección Trabajo Autónomo
2. Diseña y desarrolla una plataforma de organización de recetas de cocina. El sistema debe permitir la creación, visualización y manipulación de diferentes recetas culinarias con ingredientes y pasos de preparación.


Pasos sugeridos:

- Clase Ingrediente:  
Define una clase Ingrediente con propiedades como nombre, cantidad y unidad (por ejemplo, "azúcar", "2", "cucharadas").  
Agrega un constructor para inicializar los atributos.
- Clase PasoPreparacion:  
Define una clase PasoPreparacion con una propiedad descripción para los pasos de preparación de la receta.  
Agrega un constructor para inicializar la descripción.
- Clase Receta:  
Define una clase Receta con propiedades como nombre, categoría y una lista de objetos Ingrediente y PasoPreparacion.  
Agrega métodos para agregar ingredientes y pasos a la receta.
- Clase Main:  
En la clase principal, crea instancias de Receta y agrega ingredientes y pasos a través de métodos.  
Muestra la información de la receta incluyendo ingredientes y pasos de preparación.

#### CODIGO DE RECETA CULINARIA

```
class Ingrediente:
    def __init__(self, nombre, cantidad, unidad):
        self.nombre = nombre
        self.cantidad = cantidad
        self.unidad = unidad
```

```
class PasoPreparacion:
    def __init__(self, descripcion):
        self.descripcion = descripcion
```

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

```

class Receta:
    def __init__(self, nombre, categoria):
        self.nombre = nombre
        self.categoria = categoria
        self.ingredientes = []
        self.pasos_preparacion = []

    def agregar_ingredientes(self, nombre, cantidad, unidad):
        ingrediente = Ingrediente(nombre, cantidad, unidad)
        self.ingredientes.append(ingrediente)

    def agregar_paso_preparacion(self, descripcion):
        paso = PasoPreparacion(descripcion)
        self.pasos_preparacion.append(paso)

    def mostrar_receta(self):
        print(f"Receta: {self.nombre}")
        print(f"Categoría: {self.categoria}")
        print("\nIngredientes:")
        for ingrediente in self.ingredientes:
            print(f"- {ingrediente.cantidad} {ingrediente.unidad} de {ingrediente.nombre}")
        print("\nPasos de Preparación:")
        for i, paso in enumerate(self.pasos_preparacion, start=1):
            print(f"{i}. {paso.descripcion}")

mi_receta = Receta("Tarta de Manzana", "Postre")

mi_receta.agregar_ingredientes("Manzanas", 4, "unidades")
mi_receta.agregar_ingredientes("Azúcar", 200, "gramos")
mi_receta.agregar_ingredientes("Harina", 150, "gramos")
mi_receta.agregar_ingredientes("Canela", 1, "cucharadita")

mi_receta.agregar_paso_preparacion("Pelar y cortar las manzanas en rodajas.")
mi_receta.agregar_paso_preparacion("Mezclar las manzanas con el azúcar y la canela.")
mi_receta.agregar_paso_preparacion("Forrar un molde con la masa de harina y verter la mezcla de manzanas.")
mi_receta.agregar_paso_preparacion("Hornear a 180°C durante 40 minutos.")

mi_receta.mostrar_receta()

```


```

Categoría: Postre

Ingredientes:
- 4 unidades de Manzanas
- 200 gramos de Azúcar
- 150 gramos de Harina
- 1 cucharadita de Canela

Pasos de Preparación:
1. Pelar y cortar las manzanas en rodajas.
2. Mezclar las manzanas con el azúcar y la canela.
3. Forrar un molde con la masa de harina y verter la mezcla de manzanas.
4. Hornear a 180°C durante 40 minutos.

```

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

## Sesión 2:

- Sustente la guía de laboratorio en esta sesión, comprima y entregue a través de Canvas.

### Subtemas:

Sintaxis de las clases.  
Sintaxis de los atributos.  
Sintaxis de los métodos.

### Preguntas Orientadoras

En este apartado se realiza el análisis de los datos obtenidos, estos pueden ser de forma cualitativa o cuantitativa según la naturaleza de la práctica.

¿Cuáles fueron los aprendizajes obtenidos al realizar esta guía?, liste como mínimo 3 aprendizajes y relaciónelos con su futuro que hacer profesional.

RTA. Aprendimos el diseño y desarrollo de un programa con un nivel de dificultad más de los que anteriormente habríamos desarrollado ya que implementamos diferentes variables y métodos que no veíamos como lo fue en el organigrama para las recetas, algunas de las variables en mención son nombre, cantidad y unidad siendo propiedades de las clases, mientras que los métodos como `mostrar_info()` y `mostrar_paso()` proporcionan funcionalidad para mostrar información detallada sobre ingredientes y pasos de preparación. Estos ejemplos demuestran cómo las variables y métodos pueden utilizarse para implementar la lógica del programa y brindar interacción con los usuarios.

¿Dónde presento mayor dificultad resolviendo la guía? y ¿cómo lo resolvieron? ¿cuáles fueron las estrategias de solución?

RTA Al implementar el método `mostrar_receta()` en la clase Receta, lo resolvimos buscando El objetivo del método `mostrar_receta()`, donde utilizamos un bucle for para recorrer la lista de ingredientes y la lista de pasos de preparación. Para cada ingrediente, donde se llama al método `mostrar_info()` de la clase Ingrediente para mostrar la información detallada. Para cada paso, se llama al método `mostrar_paso()` de la clase Paso Preparación para mostrar el paso de preparación con su número correspondiente

### Presaberes Requeridos

Se requiere conocer los fundamentos de Programación e Ingeniería de Software, modelos y definiciones básicas relacionadas con el análisis, diseño, desarrollo e implementación de software.

### Marco conceptual o referencial \*


Para iniciar esta guía, se sugiere leer el artículo “Examining effectiveness of learning object-oriented programming paradigm through propriety game-based learning games. Proceedings of the European Conference on Games-Based Learning, 2016–January, 796–804. Wong, Y. S. ( 1 ), Yatim, M. H. M. ( 2 ), & Tan, W. H. ( 2 ). (n.d.). (Recuperado Base de datos Scopus).

En el desarrollo de la programación orientada a objetos se encuentran muchas clases para diferentes tipos de objetos que incluyen:

- ✓ Clases escritas por usted mismo
- ✓ Clases escritas por otros programadores
- ✓ Clases pertenecientes al propio lenguaje de programación
- ✓ Estas clases describen las propiedades de los objetos (campos) y su comportamiento (métodos).

A continuación, se presentan diferentes situaciones donde se diseñan y aplican diferentes clases para resolver algún tipo de problema y modelar los componentes involucrados

¡Es el cumpleaños de Ximena! Usted ha organizado un grupo de ocho amigos para celebrar en un restaurante local. Cuando el grupo recibe su factura, nadie está muy seguro de lo que deben. Sólo conoce el total de todos antes del IVA (5%) y propina (10%). ¡Afortunadamente! Usted trajo su computadora portátil y se les pide que escriban un programa que calcula el total de todos.

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

A continuación, se muestra una tabla donde se presenta el precio de cada plato pedido por cada persona antes del IVA (5%) y de la propina (10%)

Tabla 1: UMB. Personas y valor de consumo. 2019.

Persona 1 : \$10000	Persona 5 : \$7000
Persona 2 : \$12000	Persona 6 : \$15000 (Ximena)
Persona 3 : \$9000	Persona 7 : \$11000
Persona 4 : \$8000	Persona 8 : \$30000

Fuente Autoría propia.

### Modelado de Objetos:

Es posible que haya tenido la tentación de modelar el total de cada persona escribiendo el siguiente código:

```
import java.text.DecimalFormat;
public class PagoCuenta {
    public static void main(String[] args) {

        DecimalFormat df = new DecimalFormat("$###,###,###.##");
        double personal = 10000;
        double total1 = personal * (1 + .05 + .10);
        System.out.println("Personal : " + df.format(total1));
    }
}
```

Fuente: autoría propia

### Modelando más Objetos:

Cuando necesitaba modelar dos invitados a la cena, es posible que se sintieran tentados a copiar, pegar y cambiar el nombre:

```
import java.text.DecimalFormat;
public class PagoCuenta {
    public static void main(String[] args) {

        DecimalFormat df = new DecimalFormat("$###,###,###.##");
        double personal = 10000;
        double total1 = personal * (1 + .05 + .10);
        System.out.println("Personal : " + df.format(total1));


        double persona2 = 12000;
        double total2 = persona2 * (1 + .05 + .10);
        System.out.println("Persona2 : " + df.format(total2));
    }
}
```

Fuente: autoría propia

### Modelando muchos objetos

¿Qué pasaría si necesitara 1.000 invitados?

Usted podría pensar ... ¿De verdad tengo que copiar, pegar y renombrar 1,000 veces?

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

### Las variables ofrecen flexibilidad

Si necesita cambiar la tasa del IVA o porcentaje de propina, no necesitamos hacer 1.000 ediciones, simplemente editamos cada variable una vez.

```
double IVA = 0.05;
double propina = 0.10;

double personal = 10000;
double totall = personal * (1 + IVA + propina);
System.out.println("Personal : " + df.format(totall));

double persona2 = 12000;
double total2 = persona2 * (1 + IVA + propina);
System.out.println("Persona2 : " + df.format(totall));
```

Fuente: autoría propia

### Los métodos ofrecen una flexibilidad similar

Se repiten los mismos comportamientos matemáticos y de impresión. En su lugar, esta lógica puede escribirse en un método.

```
double IVA = 0.05;
double propina = 0.10;

double personal = 10000;
double totall = personal * (1 + IVA + propina);
System.out.println("Personal : " + df.format(totall));

double persona2 = 12000;
double total2 = persona2 * (1 + IVA + propina);
System.out.println("Persona2 : " + df.format(totall));
```

Fuente: autoría propia

### Cuando usar métodos

Es una buena idea escribir un método si

- Se encuentra repitiendo líneas de código muy similares, incluyendo cálculos
- Necesidad de describir el comportamiento de un objeto

### Cómo utilizar un método principal

- El método principal (main) se conoce como un controlador.
  - Utilizarlo para impulsar los eventos de un programa.
  - Utilizarlo para acceder a campos y métodos u otras clases.
- El método principal no describe el comportamiento de ningún objeto en particular.
  - Manténgalo separado de las clases de objetos.
  - Utilice solo un método principal para cada aplicación.

### ¿Cómo son las clases de objetos?

El código debe ajustarse al siguiente formato.

Veamos cómo podemos conseguir que nuestro código tenga esta apariencia:


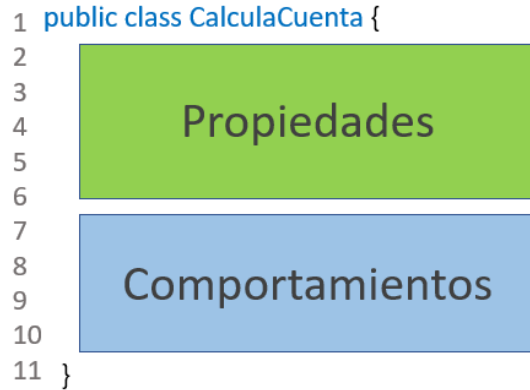
	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

Figura 1. UMB. Propiedades y Métodos. 2019.



Fuente: autoría propia

#### Paso 1:

Mover campos desde el método principal hacia el inicio de la clase principal

```

public class CalculaCuenta {
    double IVA = 0.05;
    double propina = 0.10;
    double precioOriginal = 10000;

    public static void main(String[] args) {

        //double IVA = 0.05;
        //double propina = 0.10;


        //double personal = 10000;
        double totall = personal * (1 + IVA + propina);
        System.out.println(totall);
    }
}

```

Fuente: autoría propia

#### Paso 2:

Mover comportamientos repetitivos del método principal

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

```

public class CalculaCuenta {
    // Campos o propiedades de la clase
    double IVA = 0.05;
    double propina = 0.10;
    double precioOriginal = 10000;

    //Métodos
    public void CalculaTotal(){
        // Calcula el total después de IVA y propina
        // Imprime los valores
    }

    public static void main(String[] args) {
        //double total1 = personal * (1 + IVA + propina);
        //System.out.println(total1);
    }
}

```

Fuente: autoría propia

### Paso 3:

Suprimir el método principal.

### Paso 4:

Clase Final

```

public class CalculaCuenta {
    // Campos o propiedades de la clase
    double IVA = 0.05;
    double propina = 0.10;
    double precioOriginal = 10000;

    //Métodos
    public void CalculaTotal(){
        // Calcula el total después de IVA y propina
        // Imprime los valores
    }

    //public static void main(String[] args) {
    //}
}

```

```

public class CalculaCuenta {
    // Campos o propiedades de la clase
    double IVA = 0.05;
    double propina = 0.10;
    double precioOriginal = 10000;

    //Métodos
    public void CalculaTotal(){
        // Calcula el total después de IVA y propina
        // Imprime los valores
    }
}

```

### ¿Dónde debe ir el método principal?

```

public class GestorCuenta{
    public static void main(String[] args) {

        //Crea la instancia de un objeto de la clase CalculaCuenta

        CalculaCuenta calc = new CalculaCuenta();
        calc.propina = 0.10;    // Modifica el campo
        calc.CalculaTotal();    // Llama un método
    }
}


```

El método principal se debe crear en otra clase, por ejemplo: GestorCuenta.

El método principal impulsa la acción del programa:

- Crea instancias de objetos.
- Llama los campos y los métodos de una instancia usando el operador punto (.).

### Variables para Objetos

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

```

int          edad = 22;
String       str  = "¡Feliz Cumpleaños!";
Scanner      sc   = new Scanner();
CalculaCuenta calc = new CalculaCuenta();

```

Tipo
Nombre
Valor

Fuente: autoría propia

Los objetos, como las primitivas (int, double, etc.), están representados por variables.

- La mayoría de los objetos requieren la palabra clave **new** cuando se inicializan para crear nuevas instancias.
- Esto se llama instanciación de un objeto.
- Existen algunas excepciones, como los objetos tipo String, que no requieren palabra clave new.

### Uso del operador punto

Coloque el operador punto (.) Después del nombre de una variable para acceder a sus campos o métodos.

```

public class CalculaCuenta {
    // Campos o propiedades de la clase
    double IVA = 0.05;
    double propina = 0.10;
    double precioOriginal = 10000;

    //Métodos
    public void CalculaTotal(){
        // Calcula el total después de IVA y propina
        // Imprime los valores
    }
}


public class CalculaCuenta {
    double propina = 0.15; // Inicializa la variable con 0.15
    public void imprimePropina(){
        System.out.println(propina);
    }
}

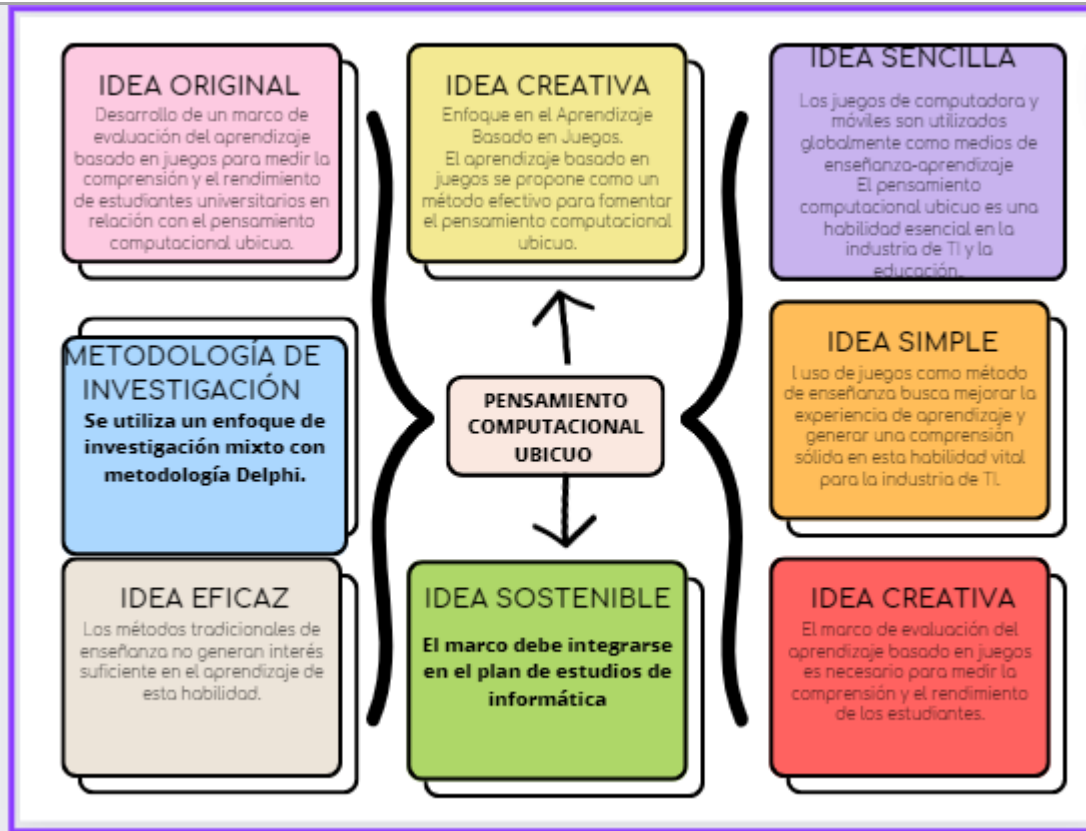
```

### Actividad de Trabajo Autónomo

Read the article: (Wong, Y. S. ( 1 ), Yatim, M. H. M. ( 2 ), & Tan, W. H. ( 2 ). (n.d.). Examining effectiveness of learning object-oriented programming paradigm through propriety game-based learning games. Proceedings of the European Conference on Games-Based Learning, 2016–January, 796–804.). (Recuperado Base de datos Scopus) and make a Mind map showing the main idea and the secondary ideas indicated by the authors. Add the conclusions of the article.



	Nombre del Proceso:	<b>GESTIÓN DE LABORATORIOS</b>	<b>CODIGO: LA-FM-001</b>
	Nombre del Documento:	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>VERSION: 7</b>
			<b>FECHA: 15/junio/2022</b>



1. Applying the concepts seen in the theoretical framework, answer the following questions:
2. How would you define the air routes class with its attributes?


To define an air route class with its attributes, we need to identify what information is represented in an air route. some common attributes could be origin, destination, distance, airline, flight number, departure time, arrival time.

The `_init_` method is the constructor of the Airway class, where attributes are assigned values using (self) which represents the current instance of the class.

3. How would the class instantiate air routes for the Sidney-Panamá object?

An instance of the air route class called `ruta_sidney_panama` is created, providing the corresponding values to the attributes in the following order:

- Origen: "sidney"
- Destino: "panama"
- Distancia: 10000 (kilometro)
- Aerolena: "aerolinea z"
- Numero de vuelo: "SP789"
- Horario de salida: "15:30"

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

- Horario de llegada: “21:45”

4. What additional methods could you create?

- Generate Detailed Report: A method that generates a detailed report of the air route, including all attributes in a readable format.
- Compare Air Routes: A method that allows two air routes to be compared to determine if they are the same or if one is shorter than the other in distance.
- Update Information: A method that allows updating the air route attributes, for example, to change the departure time or the airline.
- Data Validation: A method that performs validation of air route data, such as ensuring that distance is a positive value, or verifying that departure and arrival times are valid.

**Actividad de Comprobación del Trabajo Autónomo**

The student makes a video IN ENGLISH of a minimum of 1 and a maximum of 3 minutes exposing a summary of the main idea of the theoretical framework, the consultation of the independent work activity and the code developed in the practical example proposed in the introductory process session 1 and 2. The scholar publishes the video in the activity FORUM.

**Materiales, equipos e insumos a utilizar**


<b>Materiales, equipos e insumos proporcionados por la Universidad</b>	
<b>Ítem</b>	<b>Cantidad</b>
Computador	
Microsoft Office	
Netbeans	
Plugin de EasyUML en Netbeans	
Acceso a Internet	
Acceso a las bases de datos digitales de la Universidad.	

**Materiales del estudiante**

<b>Ítem</b>	<b>Cantidad</b>
Drive o Git/GitHub	1
Conexión a Internet en el caso de trabajar en la casa	1
Software PSeInt para diseño de diagramas de flujo. <a href="http://pseint.sourceforge.net/index.php?page=descargas.php">http://pseint.sourceforge.net/index.php?page=descargas.php</a>	1

**Precauciones, nivel de riesgo y recomendaciones a considerar**

CLASIFICACIÓN DEL RIESGO	Muy alto	Medio
	Alto	Bajo
FACTORES DE RIESGO	CÓMO MINIMIZAR LOS FACTORES DE RIESGO	
El factor de riesgo para este laboratorio está clasificado como BAJO, debido a que no se han detectado consecuencias y la eficacia del conjunto de las medidas preventivas existentes es alta. En otras palabras, el riesgo está controlado	Siga las recomendaciones y consideraciones para el uso del laboratorio y los computadores del laboratorio.	
RECOMENDACIONES, CONSIDERACIONES PARA EL USO DE MATERIAL Y EPP		
<ul style="list-style-type: none"><li>• Identificar y conocer el protocolo de seguridad de laboratorios de informática.</li><li>• No navegar en internet sin autorización del docente.</li></ul>		

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>


- No ejecutar programas sin autorización del docente.
- No instalar en los equipos Software de ninguna índole.
- No trasladar equipos de cómputo de su módulo sin autorización del personal del área.
- Cuidar sus objetos personales.
- Cada alumno tiene como responsabilidad recibir las actividades de cada clase y apropiarse del material necesario para el desarrollo de estas.
- Está prohibido el ingreso o consumo de alimentos, bebidas, chicle... dentro de la sala.
- Está prohibido el uso e ingreso de dispositivos como celulares, parlantes y memorias USB sin autorización.
- No conectar ni desconectar dispositivos como teclados, mouse o conexiones, en caso de anomalía avisar al profesor para realizar cambios o conexiones.
- El trabajo debe hacerse en silencio, evitando las reuniones o interrumpiendo las actividades de otros estudiantes.
- Cuide el buen funcionamiento del equipo que la ha sido asignado, evite cambiar configuraciones o intervenir los programas y propiedades del sistema operativo, el auxiliar de laboratorio es el único autorizado.
- Todo dispositivo (teclado o mouse) que se pierda o se dañe con intención deberá ser repuesto.
- No portar maletines o morrales, estos deben quedar depositados en los lockers destinados para ello. En caso de duda pida el respectivo candado con los auxiliares de cada laboratorio.
- No rayar mesas, sillas, paredes y equipos, cuidar el aseo y orden de su puesto de trabajo.
- Se prohíbe el ingreso o exploración de páginas no autorizadas y pornográficas, es causal de sanción y expulsión (vetado) de la sala de informática por varias sesiones.

#### CONSIDERACIONES ÉTICAS

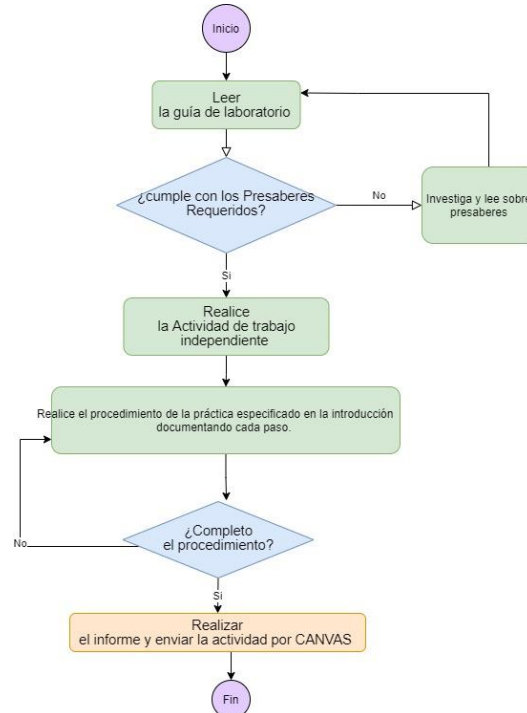
- Tener siempre presente el código de ética del ingeniero de software en especial el ítem relacionado con:
- Colegas: Cada ingeniero deberá apoyar y ser justos con los colegas, motivando a sus colegas sujetándose al código, ayudando también a su desarrollo profesional, reconocer los trabajos de otros y abstenerse a atribuirse de méritos indebidos, revisar los trabajos de manera objetiva, sincera y propiamente documentada.

#### Procedimiento y Metodología de la práctica

- El taller es grupal (en parejas).
- Cada estudiante debe ubicarse en un equipo y asegurarse de tener instaladas las aplicaciones requeridas.
- En el laboratorio el estudiante debe realizar la práctica y posteriormente puede enviar la solución del taller para ser revisada por el profesor.
- En caso de requerirse realizar la entrega de código fuente de programas comprimir como archivo .zip y anexar al correo o entrega.
- El estudiante puede formular inquietudes al profesor durante el desarrollo de la práctica.

	Nombre del Proceso:	<b>GESTIÓN DE LABORATORIOS</b>	<b>CODIGO: LA-FM-001</b>
	Nombre del Documento:	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>VERSION: 7</b>
			<b>FECHA: 15/junio/2022</b>

**Figura. No. Procedimiento de la práctica**




**Fuente: Autor**

### Criterios de Entrega – Informe de Laboratorio

1. Los solicitados en la plantilla de entrega de guía

### Criterios de Evaluación – Práctica de Laboratorio

ABILIDAD INDIVIDUAL O GRUPAL			CRITERIOS DE EVALUACIÓN					NOTA
			0 – 1,5	1,6 - 2,9	3,0 - 3,9	4,0 - 4,5	4,6 - 5,0	
competencia	Procedimental (aprender a hacer)	Utiliza herramientas CASE y aplica POO para aumentar la productividad en los procesos de desarrollo de software	Se limita a la recopilación de la información solicitada	Hace referencia a la información en la práctica	Hace relaciones básicas de la información con la práctica	Analiza la información relacionada con la práctica	Infiere la información obtenida y la relaciona con su realidad	N1
	Cognitiva (aprender a conocer)	Analiza y diseña aplicando el paradigma de Programación Orientada a Objetos	Tiene dificultades en la identificación de Clases, atributos y métodos.	Identifica algunas clases, sin embargo; no Identifica de forma adecuada los atributos y métodos	Identifica algunas clases, atributos y métodos	Identifica el concepto de clase y los atributos de la clase como también sus métodos.	Sobre una situación práctica puede identificar los conceptos de clase, atributos objetos y métodos de la clase para resolver un problema	N2
	Socio afectiva (aprender a ser)	Desarrolla habilidades de trabajo en equipo, priorizando la toma de	No hace parte del trabajo propuesto por el equipo	Parcialmente hace parte del trabajo propuesto por el equipo	Hace parte del trabajo propuesto por el equipo de acuerdo a parámetros básicos	Participa en el trabajo propuesto por el equipo de manera responsable y puntual	Participa activamente el trabajo propuesto por el equipo de manera responsable y puntual	N3

	Nombre del Proceso:	<b>GESTIÓN DE LABORATORIOS</b>	<b>CODIGO: LA-FM-001</b>
	Nombre del Documento:	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>VERSION: 7</b>
			<b>FECHA: 15/junio/2022</b>


		decisiones y la escucha de diferentes propuestas						
		Cuida, respeta y exige respeto frente a la interacción con sus pares y docentes	Frecuentemente reprocha el trabajo de sus pares y docente, y justifica sus carencias en el trabajo en grupo	A veces muestra una actitud favorable frente a la clase y se limita a responder por las condiciones básicas del trabajo	Muestra una actitud favorable frente a la clase e interactúa ocasionalmente con sus pares y docente	Muestra una buena actitud frente a la clase e interactúa con sus pares y docente	Muestra una sobresaliente actitud frente a la clase e interactúa frecuentemente con sus pares y docente	N4
	Comunicativa (aprender a convivir)	Utiliza lenguaje técnico para referirse a los diferentes conceptos que relaciona en la práctica experimental	No hace uso de un lenguaje técnico apropiado para la práctica de laboratorio	Ocasionalmente hace uso de un lenguaje técnico apropiado para la práctica de laboratorio	Hace uso de un lenguaje técnico apropiado para la práctica de laboratorio	Hace un buen uso del lenguaje técnico apropiado para la práctica de laboratorio	Hace un uso sobresaliente del lenguaje técnico apropiado para la práctica de laboratorio	N5
	Investigativa	Realiza la búsqueda bibliográfica en fuentes confiables que permitan dar respuesta a las situaciones problema evidenciados en la consulta previa y en el informe de laboratorio.	Las fuentes de información son pocas o ausentes. Si las usa son poco confiables y no contribuyen a la construcción del eje central	Las fuentes de información son restringidas o con poca diversidad. Además, no están actualizadas y contienen información poco relevante	Las fuentes de información son relevantes e informativas. Presenta los parámetros aceptables por el docente	Las fuentes de información son variadas e informativas. Adicionalmente son fiables y contribuyen al tema	Las fuentes de información son variadas y pertinentes. Además, están actualizadas y contienen información relevante al tema	N6
Nota Definitiva PROMEDIO (N1, N2, N3,N4, N5, N6)								NOTA ###

#### Palabras Clave

### PARADIGMAS DE PROGRAMACIÓN PROGRAMACION ORIENTADA OBJETOS

#### Bibliografía Recomendada

Tema	Subtema	Referente bibliográfico
Introducción a la programación orientada a objetos POO y UML.	Ejemplos de programas orientados a objetos. Lenguajes y entornos de POO. Introducción a UML: diagrama de casos de uso y diagrama de clases. Diagrama de interacción	Sznajdleder, Pablo Augusto, (2017) Programación orientada a objetos y estructura de datos a fondo: implementación de algoritmos en Java, 1er edición. Ed. Alfaomega Cloud. (Colección biblioteca UMB).
		Deitel, Paul J., (2016) Java cómo programar, Décima edición, Ed. Pearson Education. Pág. 3 (Colección biblioteca UMB).
		Joyanes Aguilar, L., & Zahonero Martínez, I. (2014). Programación en C/C++, JAVA y UML (Segunda edición). McGraw-Hill Interamericana. (Colección biblioteca UMB).
		Weitzenfeld, Alfredo. (2005). Ingeniería de software orientada a objetos con UML, Java e Internet. México: International Thomson Editores (Colección Biblioteca UMB).
		<b>LIBROS Y RECURSOS DIGITALES</b> Lassoff, M. (2017). Java Programming for Beginners. Packt Publishing. (Recuperado base de datos: EBSCO eBook Academic Collection). Castro Contreras, M. (2017). Go Design Patterns. Packt Publishing. (Recuperado base de datos: EBSCO eBook Academic Collection).

	Nombre del Proceso:	<b>CODIGO: LA-FM-001</b>
	<b>GESTIÓN DE LABORATORIOS</b>	
	Nombre del Documento:	<b>VERSION: 7</b>
	<b>FORMATO PRACTICAS DE LABORATORIOS</b>	<b>FECHA: 15/junio/2022</b>

		<p>Gamess Eric (2010). Java y la programación orientada a objetos. Colombia, South America: Universidad Icesi, Facultad de Ciencias Administrativas y Económicas, p. 51-57; Electrónico. (Recuperado Base de Datos EBSCO)</p> <p>Olier Quiceno, A. J., Gomez Salgado, A. A., &amp; Caro Pineros, M. F. (2017). Design and Implementation of a Teaching Tool for Introduction to object-oriented programming. IEEE Latin America Transactions, 15(1), 97-102 (Recuperado Base de Datos EBSCO).</p>
--	--	---

Control de cambios		
Fecha de Actualización	Descripción	Participantes
06/07/2023	Actualización formato guía de laboratorio	Olga Lucía Roa
14/08/2023	Actualización ejercicios propuestos	Diana Marcela Toquica