	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Edgar Tista Garcia
Asignatura:	Programación Orientada a Objetos.
Grupo:	Grupo 3
No de Práctica(s):	Práctica 4 – Clases y objetos.
Integrante(s):	Edwin Jaret Santiago Díaz
No. de Equipo de cómputo empleado:	Trabajo en casa.
No. de Lista o Brigada:	
Semestre:	2022 - 2
Fecha de entrega:	4 marzo 2022
Observaciones:	

CALIFICACIÓN: _____

Clases y objetos.

Objetivos

Objetivo: Aplicar los conceptos básicos de la programación orientada a objetos en un lenguaje de programación.
Objetivos de clase: Implementar una aplicación sencilla que permita entender de mejor manera los elementos del paradigma orientado a objetos.

Desarrollo

Ejemplos de la guía

Ejemplo 2. PruebaPunto.

La clase **PruebaPunto** crea una clase **Punto**, el cual guarda un valor *X* y otro valor *Y*, tiene un método que imprime los valores. En la clase **PruebaPunto** se instancia dos veces la clase **Punto**, se crean valores y después llama al método *imprimePunto* de la clase **Punto**. Este es el resultado:

```
Punto [x=5, y=8]
Punto [x=2, y=0]
```

Ahora, se implementa el método constructor para la clase **Punto**. Esto provoca que al instanciar la clase se deba de agregar sus parámetros, en este caso, el valor de *X* y de *Y*. Sin embargo, si se desea que los parámetros sean opcionales se puede crear un método constructor vacío y el otro con los atributos. Esto no altera al resultado.

Ejemplo 2. PruebaTriangulo

Se creó la clase **Triángulo** con los atributos *base* y *altura* y con 3 métodos *área()* para calcular el área del triángulo, cada uno de estos se escribió de distintas maneras para hacer una sobrecarga.

Con la clase **PruebaTriángulo** se instancia la clase **Triángulo**, se asignan valores para la *base* y *altura* y se ejecuta los 3 distintos métodos para calcular el área. Esto da de resultado:

```
Base: 5.0
Altura: 8.0
area() -> 20.0
area(6,2) -> 6.0
area(5.5f,3.2f) -> 8.8
```

Ejercicios de la clase

Ejercicio 1. Computadora

La clase **Computadora** contiene cuatro atributos, dos métodos constructores (reciben distintos parámetros) y tres métodos que se sobrecargan (imprimen un mensaje en la pantalla). Dentro de la misma clase se utiliza el método **main** para instanciar dos veces la clase **Computador,** cada instancia utiliza un método constructor y utilizan cada método sobrecargado. Este es el resultado:

```
Hola desde el futuro
Hola desde el futuro
Para: EdwinSan
```

```
Escoge una de las siguientes opciones:
1.- Crear una computadora
2.- Modificar una computadora
3.- Ver las computadoras guardadas
4.- Eliminar una computadora
```

Después, se creó la clase **MenuComputadora** en donde se implementó un menú para que el usuario tenga la opción de ejecutar los siguientes métodos:

- Crear computadora nueva: crea un objeto computadora, se le solicita al usuario por el nombre de la computadora y el procesador. Al final se agrega el objeto en un *ArrayList*.

```
1

Cual es el nombre/modelo de la comptuadora?
Asus Gaming
Que procesador tiene?
Corei 5 10ma gen
```

- Modificar computadora existente: Se valida que haya computadoras existentes, si fuera el caso se imprime la lista de las computadoras existentes con su índice y se le solicita el índice que desea modificar para después solicitarle el nombre y el procesador de la computadora nueva, si la lista está vacía se imprime un mensaje informando que no hay computadoras.

```
2

Â¿Que computadora desea modificar?

Estas son las computadoras guardadas:
Indice: 0, nombre: Asus Gaming, procesador: Corei 5 10ma gen
Ingrese el indice
0
Ingrese el nuevo nombre de la computadora
Ingrese el nuevo procesador de la comptuadora
Corei 7
```

- Ver computadoras existentes: Valida la existencia de computadoras en la *ArrayList*, si hay computadoras existentes, se imprime el índice, el nombre y el procesador que tiene cada computadora, si la lista está vacía se informa que no hay computadoras generadas.

```
3

Estas son las computadoras guardadas:
Indice: 0, nombre: , procesador: Corei 7
```

- Eliminar una computadora: Imprime la lista de las computadoras existentes, si no hay, se le informa al usuario. Al ver la lista, se le solicita al usuario el índice de la computadora que desea eliminar y se elimina.

```
4
Indice: 0, nombre: , procesador: Corei 7

Â¿Que computadora desaas eliminar? Ingrese el indice:
0
```

Un problema que tengo con mi programa que no fue posible solucionarlo fue en el método *modificarComputadora()*, después de solicitar el índice de la computadora a modificar, está asignado que se guarde el nombre de la computadora y el procesador, sin embargo al solicitarle dicha información omite la capturación del nombre de la computadora.

Este es el código comentado:

```java
// Obtiene la computadora a traves del indice
Computadora modificable = inventario.get(indice);

// Cambia el nombre de la computadora
System.out.println("Ingrese el nuevo nombre de la computadora");
String nuevoNombre = sc.nextLine();

// Cambia el procesador de la computadora
System.out.println("Ingrese el nuevo procesador de la comptuadora");
String nuevoProcesador = sc.nextLine();

// Se asignan los valores nuevos a la computadora que se quiere modificar
modificable.setProcesadorComputadora(nuevoProcesador);
modificable.setNombreComputadora(nuevoNombre);
```

Y este es en el momento de ejecutar el programa:

```
Escoge una de las siguientes opciones:
1.- Crear una computadora
2.- Modificar una computadora
3.- Ver las computadoras guardadas
4.- Eliminar una computadora
2

Â¿Que computadora desea modificar?

Estas son las computadoras guardadas:
Indice: 0, nombre: Asus gaming, procesador: Corei 5
Indice: 1, nombre: Asus Gaming, procesador: Corei 5 10ma generacion
Indice: 2, nombre: Lenovo, procesador: Amd
Ingrese el indice
1
Ingrese el nuevo nombre de la computadora
Ingrese el nuevo procesador de la comptuadora
Corei 7
```

El procesador si se cambió con éxito pero el nombre no:

```
Escoge una de las siguientes opciones:
1.- Crear una computadora
2.- Modificar una computadora
3.- Ver las computadoras guardadas
4.- Eliminar una computadora
3

Estas son las computadoras guardadas:
Indice: 0, nombre: Asus gaming, procesador: Corei 5
Indice: 1, nombre: , procesador: Corei 7
Indice: 2, nombre: Lenovo, procesador: Amd
```

Para guardar las computadoras creadas se ocupa de colección la lista *ArrayList* para que esta pueda ser modificada en su tamaño y elementos, además recibe la clase **Computadora**. Se creó de esta manera:

```
1  static ArrayList<Computadora> inventario = new ArrayList<>();
```

Se creó con la palabra **static** para que pueda ser accedida a través de la clase y no ser necesario instanciar un objeto al igual que los métodos *crearComputadora* y *eliminarComputadora*.

En el caso de los métodos setNombreComputadora y setProcesadorComputadora se utiliza las referencias **this** , reciben un parámetro y los atributos que se modifican son privados.

Conclusiones

Se debe de evitar la sobrecarga en los métodos constructores y en los métodos, en el caso de los métodos constructores se podría crear un constructor vacío y otro con los parámetros deseados, pero es una mala práctica. Se debe de definir claramente la estructura de las clases.

El manejo de *ArrayList* para la modificación de una lista es la opción ideal. El uso de **static** debe de usarse para el manejo de variables y métodos de clase para que no se instancia un objeto innecesariamente.