Universidad Autónoma de Baja California

FACULTAD DE CIENCIAS QUÍMICAS E INGENIERÍA

FCQI

Reporte de actividad

Programación orientada a objetos.

Práctica No. 5:

Investigación y práctica sobre la clase String en Java

Alumno

Joshua Osorio Osorio - 1293271

Docente

Itzel Barriba Cazares

3 de marzo de 2025

ÍNDICE

Objetivo

Los estudiantes implementarán asociaciones unidireccionales y bidireccionales en Java, comprendiendo cómo se establecen las relaciones entre clases.

Introducción

- Lee la teoría sobre asociaciones en Java. Diferencia entre asociación unidireccional y bidireccional.
- Implementa los ejemplos propuestos a continuación en Java.
- Ejecuta y analiza los resultados

Para llevar a cabo esta investigación y práctica, se requirió lo siguiente.

Lista de materiales

- Computadora con entorno de desarrollo compatible con Java.
- JDK (Java Development Kit) instalado (versión 8 o superior recomendada).
- Editor de texto.
- Documentación oficial de Java para referencia sobre la clase String.
- Recursos de apoyo como libros y artículos en línea sobre String en Java.
- Compilador de Java (incluido en el JDK) para ejecutar y probar el código.

Desarrollo

Programa 1:

Un sistema de comercio electrónico maneja información sobre clientes y pedidos. Un cliente puede realizar múltiples pedidos, pero cada pedido pertenece a un solo cliente.
• Identifica las clases y sus atributos esenciales
• Define los métodos necesarios para gestionar la relación entre clientes y pedidos. • Dibuja un diagrama de clases donde represente la asociación entre Cliente y Pedido. • Piensa en cómo se representa esta relación en código. ¿Usarás una lista de pedidos en la clase Cliente? ¿Cómo accederás a la información de un pedido?
Después de analizar el problema, implementa las clases en código y prueba su funcionalidad creando cliente y pedidos.

Pedido.java

```java
public class Pedido {

    private String id;
    private int noSegimiento;
    private String direccion;
    private float peso;
    private float largo;
    private float ancho;
    private float alto;

    public Pedido() {
        this.alto = 0;
        this.ancho = 0;
        this.direccion = "No especificado";
        this.id = "No especificado";
        this.largo = 0;
        this.noSegimiento = 0;
        this.peso = 0;
    }

    public Pedido(String id, int noSegimiento, String direccion, float peso,
float ancho, float largo, float alto) {
        this.alto = alto;
```

```
        this.ancho = ancho;
        this.direccion = direccion;
        this.id = id;
        this.largo = largo;
        this.noSegimiento = noSegimiento;
        this.peso = peso;
    }

    // Metodos setters
    public void setId(String id) {
        this.id = id;
    }

    public void setNoSegimiento(int noSegimiento) {
        this.noSegimiento = noSegimiento;
    }

    public void setDireccion(String direccion) {
        this.direccion = direccion;
    }

    public void setPeso(float peso) {
        this.peso = peso;
    }

    //Metodos getter.
    public String getId() {
        return id;
    }

    public int getNoSegimiento() {
        return noSegimiento;
    }

    public String getDireccion() {
        return direccion;
    }

    public float getPeso() {
        return peso;
    }

    public float getLargo() {
        return largo;
```

```
    }

    public void setLargo(float largo) {
        this.largo = largo;
    }

    public float getAncho() {
        return ancho;
    }

    public void setAncho(float ancho) {
        this.ancho = ancho;
    }

    public float getAlto() {
        return alto;
    }

    public void setAlto(float alto) {
        this.alto = alto;
    }

    public String toString() {
        return "\nid:\t" + id + "\nNo seguimiento:\t" + noSegimiento +
"\nDireccion:\t" + direccion + "\nPeso:\t" + peso + "\nLargo:\t" + largo +
"\nAncho:\t" + ancho + "\nAlto:\t" + alto;
    }

}
```

Cliente.java

Java

```
public class Cliente {

    private String nombre;
    private String direccion;
    private String Celular;
    private Pedido pedido;

    // constructor.
```

```java
public Cliente() {
    this.Celular = "No especificado";
    this.direccion = "No especificado";
    this.nombre = "No especificado";
}

public Cliente(String Celular, String direccion, String nombre) {
    this.Celular = Celular;
    this.direccion = direccion;
    this.nombre = nombre;
}

// metodo Getter.
public String getCelular() {
    return Celular;
}

public String getDireccion() {
    return direccion;
}

public String getNombre() {
    return nombre;
}

public Pedido getPedido() {
    return pedido;
}

// Metodos Setter
public void setCelular(String Celular) {
    this.Celular = Celular;
}

public void setNombre(String nombre) {
    this.nombre = nombre;
}

public void setDireccion(String direccion) {
    this.direccion = direccion;
}

public void setPedido(Pedido pedido) {
    this.pedido = pedido;
```

```
    }

    public String toString() {
        return "\nNombre:\t" + nombre + "\nDirreccion:\t" + direccion +
"\nCelular:\t" + Celular + "\n";
    }

}
```

Principal.java

```Java
/*-----------------------------------
Practica: 6 Gestión y manipulación de arreglos de objetos en Java.
Nombre: Joshua Osorio
Materia: Programación Orientada a Objetos
Fecha: Marzo/20/2025
-----------------------------------*/

import java.util.Scanner;

public class Principal {

    public static void main(String[] args) {

        final int SIZE_ARR_PEDIDO = 4; // Size of array.

        Scanner input = new Scanner(System.in);

        // Creación de un objeto de
        Cliente[] bdCliente = new Cliente[SIZE_ARR_PEDIDO];
        // Pedido pedido = new Pedido();

        // Declaración de las variables utilizadas en el programa
        String id, direccion, nombre, celular, texto;
        int opcion, noSegimiento, numCliente = 0;
        float peso, largo, ancho, alto;

        System.out.println("Sistema para capturar pedidos");

        // Inicio el ciclo do while para estar mostrando siempre las opciones.
        do {
```

```
            // Se despliega información para mostrar las opciones.
            System.out.println("\nMenú");
            System.out.println("\t1 - Agregar bdCliente");
            System.out.println("\t2 - Agregar pedido");
            System.out.println("\t3 - Mostrar Lista de clientes");
            System.out.println("\t0 - Salir");

            // Se captura la opción que desea.
            System.out.print("\tOpción: ");
            opcion = input.nextInt();

            // Limpia el búfer después de ingresar un numero
            input.nextLine();

            switch (opcion) {
                case 1:
                    System.out.println("\tCrear cliente");
                    System.out.printf("\tNombre:\t");
                    nombre = input.nextLine();
                    System.out.printf("\tDirección:\t");
                    direccion = input.nextLine();
                    System.out.printf("\tCelular:\t");
                    celular = input.next();
                    input.nextLine();

                    bdCliente[numCliente] = new Cliente(celular, direccion,
nombre);
                    texto = bdCliente[numCliente].toString();
                    System.out.printf(texto);
                    break;
                case 2:
                    System.out.println("\tAgregar pedido");
                    System.out.printf("\tId (texto):\t");
                    id = input.next();
                    input.nextLine();
                    System.out.printf("\tNumero de segimiento (numero):\t");
                    noSegimiento = input.nextInt();
                    input.nextLine();
                    System.out.printf("\tPeso:\t");
                    peso = input.nextFloat();
                    input.nextLine();
                    System.out.printf("\tLargo:\t");
                    largo = input.nextFloat();
                    input.nextLine();
```

```
                    System.out.printf("\tAncho:\t");
                    ancho = input.nextFloat();
                    input.nextLine();
                    System.out.printf("\tAlto:\t");
                    alto = input.nextFloat();
                    input.nextLine();

                    bdCliente[numCliente].setPedido(new Pedido(id,
noSegimiento, bdCliente[numCliente].getDireccion(), peso, ancho, largo, alto));

                    texto = bdCliente[numCliente].getPedido().toString();
                    System.out.printf(texto);
                    numCliente += 1;
                    break;
                case 3:
                    System.out.println("\tMostrar clientes y pedidos.");
                    mostrarClientes(bdCliente);
                    break;
                case 0:
                    System.out.println("\tSaliendo del sistema...");
                    break;

                default:
                    System.out.println("\tOpción no válida, intente
nuevamente.");
                    break;

            }
            System.out.println("Presiona cualquier tecla para continuar...");
            input.next();
            System.out.flush();
        } while (opcion != 0);

        input.close();
    }

    // Método auxiliar para mostrar los pedidos
    public static void mostrarClientes(Cliente[] clientes) {
        String texto;
        int numC = 0;

        if (clientes == null || clientes.length == 0) {
            System.out.println("\tNo se encontraron clientes.");
            return;
```

```
            }

            System.out.println("\tClientes encontrados:");
            for (Cliente cliente : clientes) {
                if (cliente != null) {
                    System.out.println("Cliente No:\t" + numC);

                    texto = cliente.toString();
                    System.out.println(texto + "\n");
                    texto = cliente.getPedido().toString();
                    System.out.println(texto + "\n");
                }
                numC += 1;
            }
        }
    }
```

Salida en terminal del programa 1:

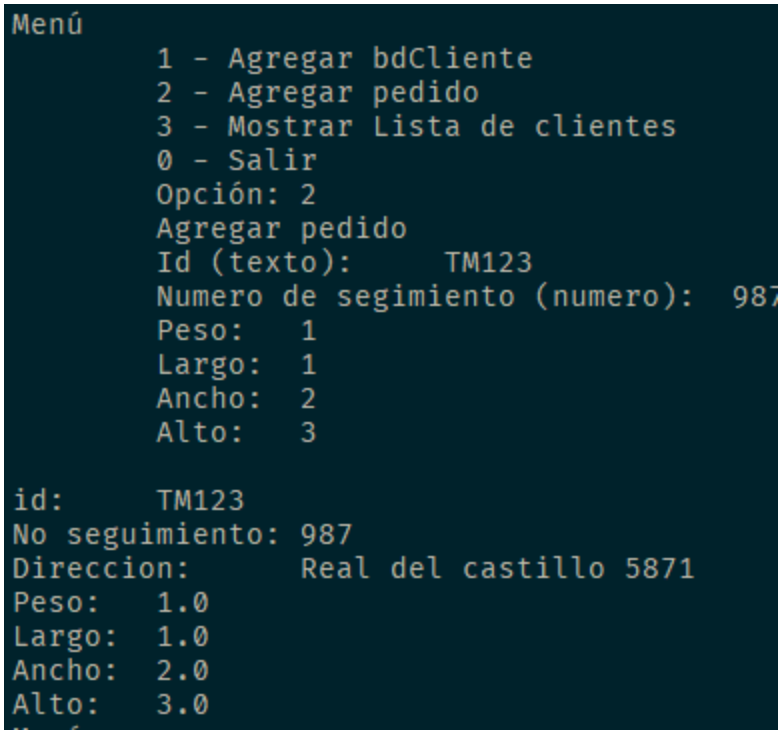
```
C:\Users\Okuyt\Desktop\POO\Taller\Taller7\Prog1>javac Cliente.java Pedido.java Principal.java

C:\Users\Okuyt\Desktop\POO\Taller\Taller7\Prog1>java Principal.java
Sistema para capturar pedidos

Menú
        1 - Agregar bdCliente
        2 - Agregar pedido
        3 - Mostrar Lista de clientes
        0 - Salir
        Opción:
```

```
        Opción: 1
        Crear cliente
        Nombre: Joshua Osorio O.
        Dirección:      Real del castillo 5871
        Celular:        6646053294

Nombre: Joshua Osorio O.
Dirreccion:     Real del castillo 5871
Celular:        6646053294
```



Programa 2:

En una universidad, cada profesor está asignado a un departamento, y cada departamento tiene un único profesor a cargo.

• Análisis el problema e identifica las clases. ¿Qué atributos son esenciales para Profesor y Departamento?

• Define cómo establecerás la relación entre profesor y departamento. ¿Cada objeto tendrá una referencia al otro?

Conclusiones

Durante esta práctica, se investigaron las características de la clase String en Java y se comprendió su inmutabilidad, lo que implica que sus objetos no pueden ser modificados

directamente. Además, se analizaron las diferencias con StringBuilder y StringBuffer, destacando sus usos en optimización de memoria y concurrencia.

Se implementó un programa en Java que demuestra la manipulación de cadenas mediante diversos métodos de String, permitiendo obtener su longitud, convertir a mayúsculas/minúsculas, contar caracteres, verificar prefijos y sufijos, y realizar reemplazos.

Durante la práctica, surgieron problemas al ingresar texto después de capturar un número, lo cual se solucionó utilizando nextLine() para limpiar el buffer. También se observó que el uso de clases facilitó la organización del código.

Este ejercicio permitió reforzar los conocimientos sobre String y su aplicación en situaciones prácticas.

Bibliografía

[1] "Java a fondo," *Google Books*. [Online]. Available: https://books.google.com.mx/books?hl=es&lr=&id=WcL2DQAAQBAJ&oi=fnd&pg=PA1&dq=%C2%BFQu%C3%A9+es+la+clase+String+en+Java%3F&ots=iRwaAyCxI7&sig=Y9QsnqLumorAhget_jn3QCjEE7w#v=onepage&q=%C2%BFQu%C3%A9%20es%20la%20clase%20String%20en%20Java%3F&f=false.

[2] I. González Pérez, A. J. Sánchez Martín, D. Vicente Hernández, y Departamento de Informática y Automática Universidad de Salamanca, «Java Threads (Hilos en Java)», *Google Scholar*, p. 1, may 2002, Disponible en: http://lsi.vc.ehu.es/pablogn/docencia/ISO/7%20Gest%20de%20Procesos/HilosJavaUSAL.pdf

[3] Dotnet-Bot, "String Class (Java.Lang)." [Online]. Available: https://learn.microsoft.com/en-us/dotnet/api/java.lang.string?view=net-android-35.0.