UNVIERSIDAD PRIVADA" FRANZ TAMAYO" - SEDE EL ALTO

FACULTAD DE INGENIERIA

CARRERA DE INGENIERIA DE SISTEMAS

ESTRUCTURA DE DATOS

BIBLIOTECA

DOCENTE : Ing. WILLIAM RODDY BARRA PAREDES

ESTUDIANTES : CALLE VAQUIATA MARCO ANTONIO

LEON LUIS JOSIAS JONATHAN

QUIROGA HUARISTE ANDRES VLADIMIR

SARZO LAURA ILIA ARACELI

VELASCO ARUQUIPA IRIS MICHELLE

FECHA : 08 DE DICIEMB|RE DE 2022

El Alto, La Paz-Bolivia

2022

INDICE

CAPITULO I

INTRODUCCIÓN

Las estructuras de datos son una forma de organizar los datos en el ordenador para que pueda ser utilizados de manera eficiente. En este trabajo realizado aplicaremos los temas que aprendimos durante el presente semestre, siendo esta nuestra elección para la creación de una Biblioteca de una universidad. Hicimos el uso del entorno de desarrollo para la creación de software informática (INTELLI IDEA), se mostrará el modelo lógico, sus clases y sus diferentes comandos.

1.2 PROBLEMA GENERAL

El problema a la que se enfrenta la biblioteca es el orden de los libros cuando se requiere algún tipo de información para el bibliotecario para agilizar las búsquedas de libros ya sea que solicite el administrador.

Debido a que no cuenta con un sistema automatizado que ayude a tener acceso a la información

oportuna, esto produce resultados negativos al factor social, económico y tecnológico. Es por aquello que la biblioteca necesita tener dicha información disponible de manera rápida y precisa.

1.3 OBJETIVOS

1.3.1 Objetivo General

Implementar un sistema de información bibliotecario que permita tener una información de los

libros de forma rápida y eficiente.

1.3.2 Objetivo Especifico

Desarrollar un programa en INTELLI IDEA para facilitar la información de los libros de la biblioteca.

Organizar y proporcionar información de manera sencilla.

Mediante este sistema obtener la información más rápida, tendrá la biblioteca una administración organizada y clara ante el uso de datos del cliente como también el de los libros.

Otro de los objetivos del programa es mantener la calidad e integridad de los datos bajo cualquier circunstancia.

De esta manera el bibliotecario tendrá de manera digital y/o virtual de una biblioteca mas organizada con respecto al manejo de los libros.

Tener flexibilidad al momento de modificar o ingresar datos que son cambiantes de forma continua, esto hará que nos mantengamos al día con la información de manera simple.

CAPITULO II

MARCO TEORICO

Estructura de Datos

Las estructuras de datos en programación son un modo de representar información en una computadora.

¿Para qué sirven las Estructuras de Datos?

En el ámbito de la informática, las estructuras de datos son aquellas que nos permiten, como desarrolladores, organizar la información de manera eficiente y en definitiva diseñar la solución correcta para un determinado problema.

Ya sean las más utilizadas comúnmente como las variables, arrays, conjuntos o clases o las diseñadas para un propósito especifico árboles, grafos, tablas, etc. Una estructura de datos nos permite trabajar en un algo nivel de abstracción almacenado información para luego acceder a ella, modificarla y manipularla.

Tipos de Estructura de Datos

Las estructuras de datos estáticos son aquellas en las que el tamaño ocupado en memoria se define antes de que el programa se ejecute y no puede modificarse dicho tamaño durante la ejecución del programa, mientras que una estructura de datos dinámica es aquella en la que el tamaño ocupado en memoria puede modificarse durante la ejecución del programa.

Cada tipo de estructura dependerá del tipo de aplicación que se requiera. Una típica dentro de las estructuras de datos estáticos son los arrays.

Java

Java es un lenguaje de programación ampliamente utilizado para codificar aplicaciones web. Ha sido una opción popular entre los desarrolladores durante más de dos décadas, con millones

de aplicaciones Java en uso en la actualidad. Java es un lenguaje multiplataforma, orientado a objetos y centrado en la red que se puede utilizar como una plataforma en sí mismo. Es un lenguaje de programación rápido, seguro y confiable para codificarlo todo, desde aplicaciones móviles y software empresarial hasta aplicaciones de macrodatos y tecnologías del servidor.

Arrays

Un array es un tipo de dato estructurado que permite almacenar un conjunto de datos homogéneo y ordenado, todos ellos del mismo tipo y relacionados. Su condición de homogéneo, indica que sus elementos están compuestos por el mismo tipo de dato, y su condición de ordenado hace que se pueda identificar del primer al ultimo elemento que lo compone.

String

Un String o cadena de caracteres es un tipo de dato que se utiliza para almacenar texto.

l esquema general es el siguiente: String nombre="cadena"; Cuando el compilador encuentra la siguente cadena, crea un objeto String cuyo valor es Hola Mundo. También se pueden crear objetos String como se haría con cualquier otro objeto Java: utilizando new. String s = new String ("Hola Mundo.");

Integer

Un numero entero, en el contexto de la programación de computadoras, es un tipo de datos utilizado para representar números reales que tienen valores fraccionarios.

Java Integer es la clase del lenguaje Java que nos permite convertir un tipo básico int en un objeto. Esta clase contiene varios métodos estáticos que permiten realizar conversiones comunes de una forma rápida entre int e Integer o entre Integer y String

Private

Una clase, función o variable que este indicada como Private indica que solo esta accesible a los métodos de la misma clase en la que la variable o método se ha declarado.

El campo o método sólo es visible dentro de la clase donde se define. protected: El campo o método es visible en la clase en donde se define y en cualquiera de sus subclases. public: El campo o método es visible en cualquier clase.

Public Static Void

El método public static Void es un método especial, es el que permite ejecutar el código de un programa o aplicación.

```
import java.time.*;
class Main {
 public static void main(String[] args) {
  LocalDate date = LocalDate.parse("2018-05-05");
  System.out.println(date);
 }
```

While

Se utiliza la sentencia While para ejecutar en bucle en conjunto de instrucciones hasta que se cumpla una condición determinada.

Es una estructura iterativa indeterminada, este bucle comienza evaluando una condición booleana dada. Si la condición es verdadera, se ejecutan las sentencias en el cuerpo interno del bucle. Si no, pasa a la siguiente instrucción del programa. La sintaxis en general es: while (condición) {instrucciones a ejecutarse} donde condición es una expresión que da un resultado

true o false en base al cual el bucle se ejecuta o no. Escribe y prueba el siguiente código, donde además vemos un ejemplo de uso de la instrucción break;.

For

Se usa cuando queremos repetir un conjunto de instrucciones un numero finito de veces.

La instrucción for permite repetir una instrucción o una instrucción compuesta un número especificado de veces. El cuerpo de una instrucción for se ejecuta cero o más veces hasta que una condición opcional sea false. Bucles for en Java. Los bucles son un tipo de sentencia y estructura de control de flujo dentro de un programa. Se utilizan para repetir la ejecución de un bloque de código, hasta que la condición booleana asignada al bucle deja de cumplirse.

Else

Es una instrucción alternativa que se ejecuta si el resultado de una condición de prueba previa se evalúa como falsa.

Bucles for en Java. Los bucles son un tipo de sentencia y estructura de control de flujo dentro de un programa. Se utilizan para repetir la ejecución de un bloque de código, hasta que la condición booleana asignada al bucle deja de cumplirse. La función IF THEN ELSE prueba una condición, luego devuelve un valor basado en el resultado de esa condición. La expresión IF THEN ELSE puede definirse de dos maneras: IF (condición booleana) THEN (valor verdadero) ELSE (valor falso) ENDIF: el resultado devuelto dependerá de si la condición se cumple o no.

If

Un If se utiliza para evaluar una expresión condicional, si se cumple la condición (es verdadera), ejecutara un bloque de código, si es falsa es posible ejecutar otras sentencias.

La estructura condicional más simple en Java es el if, se evalúa una condición y en caso de que se cumpla se ejecuta el contenido entre las llaves {} o en caso de que se omitan se ejecuta el

código hasta el primer «;» por lo tanto si no se usan los {} la condición aplica solo a la siguiente instrucción al if.

Constructor

Es un elemento de una clase cuyo identificador coincide con el de la clase correspondiente y que tiene por objetivo obligar a controlar como se inicializa una instancia de una determinada clase.

El constructor es un tipo específico de método que siempre tiene el mismo nombre que la clase y se utiliza para construir objetos de esa clase. No tiene tipo de dato específico de retorno, ni siquiera void. Esto se debe a que el tipo específico que debe devolver un constructor de clase es el propio tipo de la clase.

Setter

Su función permite brindar acceso a propiedades especificas para poder asignar un valor fuera de clase.

Los Setters y Getters son métodos de acceso a los campos/atributos de una clase. Setters: Del Inglés Set, que significa establecer, sirven para asignar un valor a un campo/atributo. Getters: Del Inglés Get, que significa obtener, sirven para recuperar el valor de un campo/atributo

Getter

Su función es permitir el obtener el valor de una propiedad de la clase y así poder utilizar dicho valor en diferentes métodos.

Los métodos que permiten acceder al valor de un atributo se denominan "getters" (del verbo inglés "get", obtener) y los que fijan el valor de un atributo se denominan "setters" (del verbo inglés "set", fijar).

Main

El método Main es el punto de entrada de un programa ejecutable, es donde se inicia y finaliza el control del programa.

El método Main es el punto de entrada de un programa ejecutable; es donde se inicia y finaliza el control del programa. Main se declara dentro de una clase o estructura. El valor de Main debe ser static y no public.

Package

Los packages (paquetes) son una forma de organizar grupos de clases. Un paquete contiene un conjunto de clases relacionadas bien por finalidad, por ámbito o por herencia.

Los paquetes son una forma de organizar grupos de clases. Un paquete contiene un conjunto de clases relacionadas bien por finalidad, por ámbito o por herencia. Los paquetes resuelven el problema del conflicto entre los nombres de las clases. Un paquete es un contenedor de clases, que se usa para mantener el espacio de nombres de clase, dividido en compartimentos. import nombre-paquete; Permiten restringir la visibilidad de las clases que contiene: Se pueden definir clases en un paquete sin que el mundo exterior sepa que están allí.

Clase

Son plantillas para la creación de objetos, la clase forma la base para la programación orientada a objetos en java.

Las clases en Java (Java Class) son plantillas para la creación de objetos. Como tal, la clase forma la base para la programación orientada a objetos en Java, la cual es una de los principales paradigmas de desarrollo de software en la actualidad. La clase es la definición general de una entidad sobre la que estamos interesados en realizar algún tipo de procesamiento informático.

Ejemplo: personas, coches, libros, alumnos, productos. La clase es la base de la PDO en Java. Objeto: Elemento real asociado a una clase (instancia de una clase).

This

This hace referencia al objeto actual de la clase. This hace referencia al objeto actual de la clase, es decir, a una instancia concreta de la clase y nos sirve para usar los métodos y atributos de esa clase desde alguno de sus métodos, para llamar a otro de sus constructores o simplemente para pasarle el objeto completo a algún otro método u objeto.

La utilización de this en el tercer constructor de la clase, permite referirse directamente al objeto en sí, en lugar de permitir que el ámbito actual defina la resolución de variables, al utilizar como parámetro formal y después this para acceder a la variable de instancia del objeto actual.

Boolean

La mayoría de los lenguajes posee un tipo de datos para almacenar valores de verdad, en java este tipo de datos se llaman Boolean y puede almacenar únicamente dos valores: verdadero y falso.

Una variable de tipo booleano es aquella cuyo valor es uno de otros los valores true o false. Una variable de tipo enumerado es aquella cuyo valor es uno de entre los definidos por el programador (extiende a las variables booleanas). La mayoría de los lenguajes posee un tipo de datos para almacenar valores de verdad. En Java este tipo de datos se llama boolean y puede almacenar únicamente dos valores: verdadero o falso. Constantes: true: representa el valor verdadero.

Return

La sentencia return finaliza la ejecución de la función y especifica un valor para ser devuelto a quien llama a la función.

La sentencia return se emplea para salir de la secuencia de ejecución de las sentencias de un método y, opcionalmente, devolver un valor. Tras la salida del método se vuelve a la secuencia de ejecución del programa al lugar de llamada de dicho método.

Si utilizamos return como sentencia de finalización de un método, declaramos el mismo de manera convencional con la palabra void, aquí tengamos en cuenta que, si se intenta devolver un valor desde un método declarado void, aparecerá un error de compilación.

CAPITULO III

MARCO APLICATIVO

3.1 Análisis y Diseño del sistema utilizando estructura de datos

a. Nombre del Proyecto (Sistema)

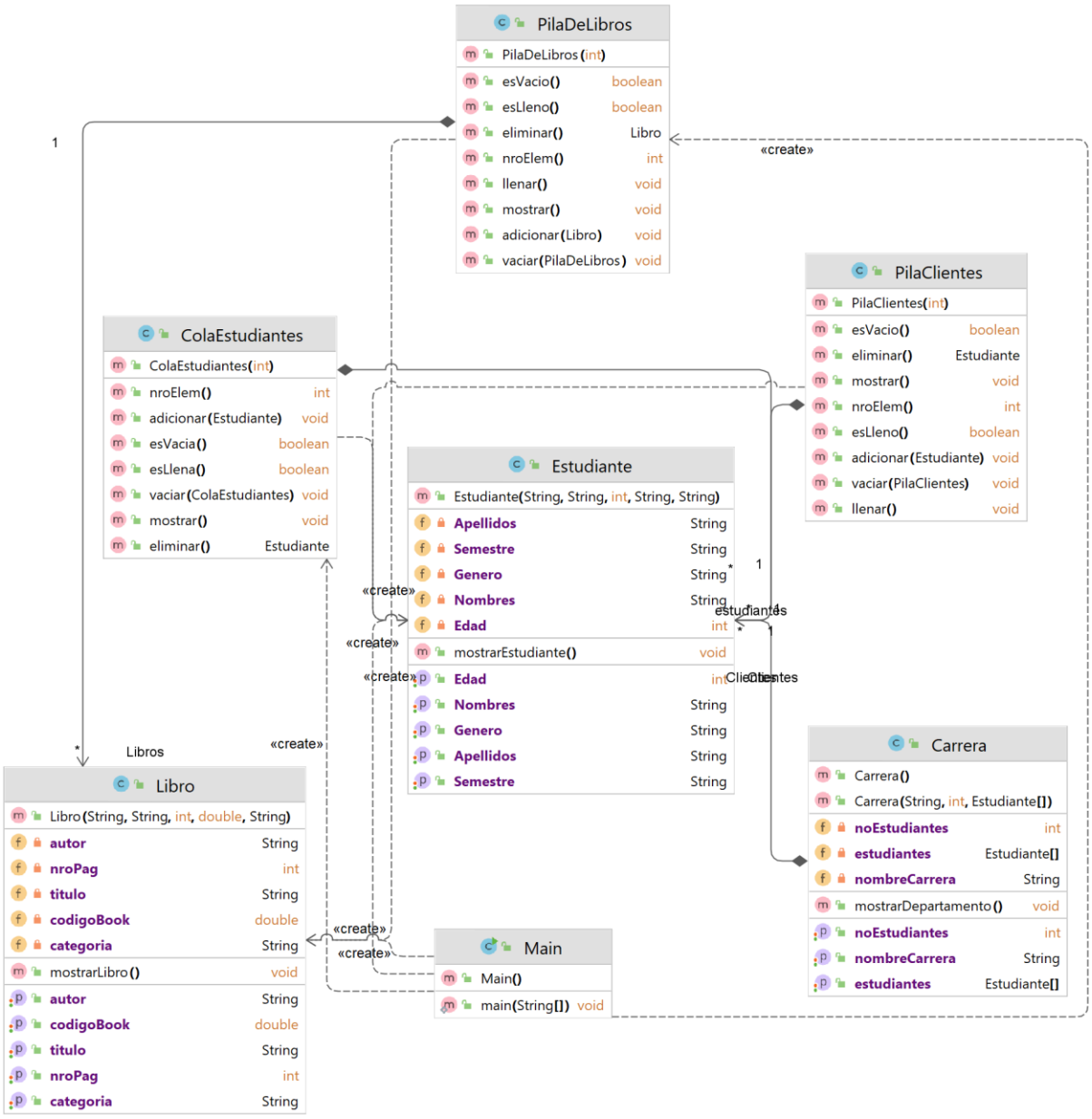
Dada la situación de mejorar la Biblioteca, identificamos que el nombre adecuado para la base de datos deberá ser Proyecto Biblioteca.

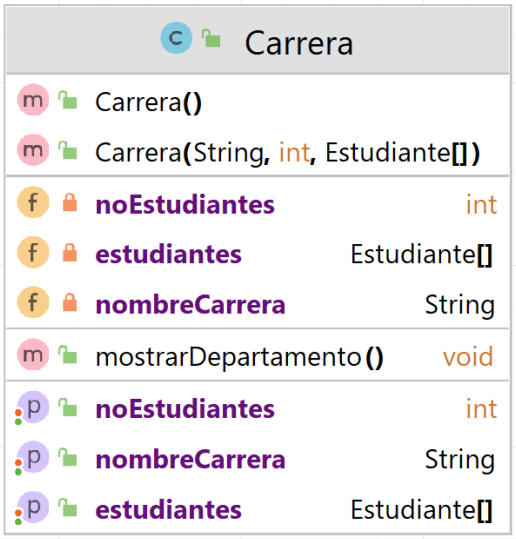
b. Entidades/ clases del sistema

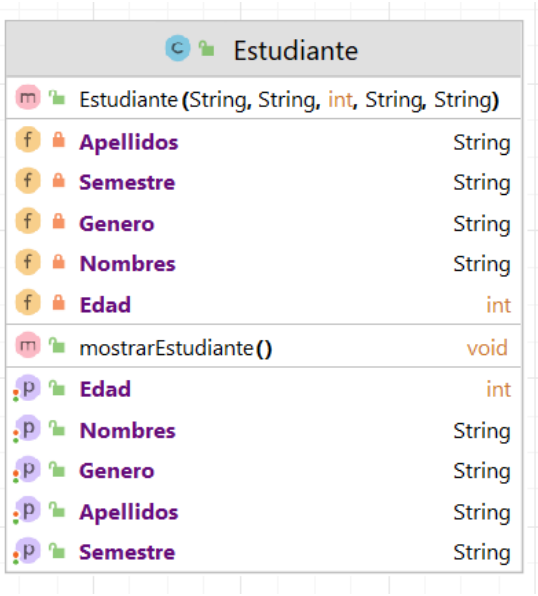
Estudiantes	Almacena datos de estudiantes
ColaEstudiantes	Almacena constructores para la cola estudiantes
Libro	Almacena los datos del libro
PilaDeLibros	Almacena para la pila de libros
PilaDeCliente	Almacene para la pila cliente
Carrera	Almacena los datos de la carrera
Main	Almacena los datos almacenados en colas, pilas y métodos

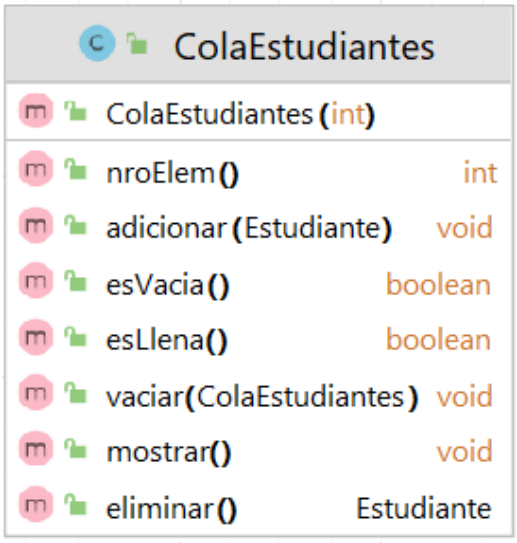
Nombre del proyecto (Sistema): ProyectoBiblioteca

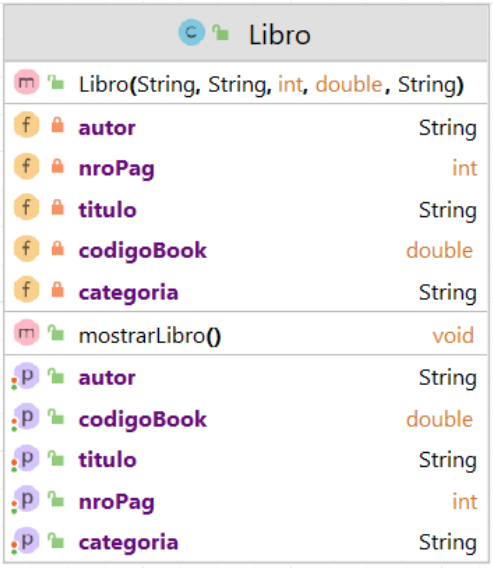
Diseño de Clases

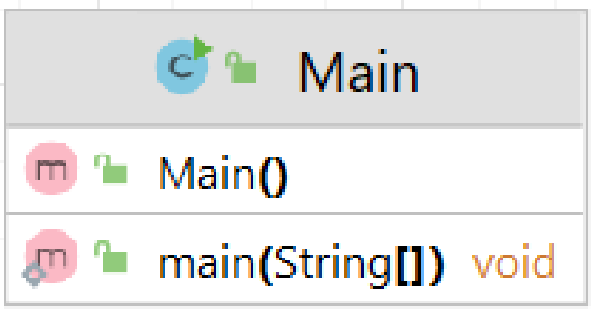


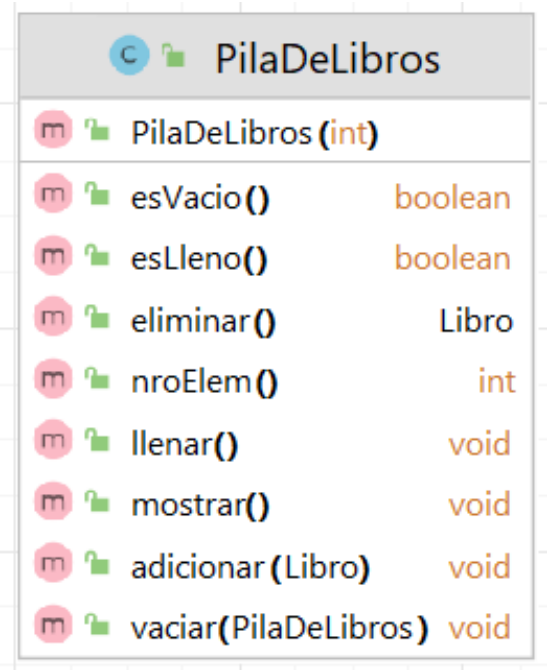

Carrera
Carrera()
Carrera(String, int, Estudiante[])
noEstudiantes int
estudiantes Estudiante[]
nombreCarrera String
mostrarDepartamento() void
noEstudiantes int
nombreCarrera String
estudiantes Estudiante[]

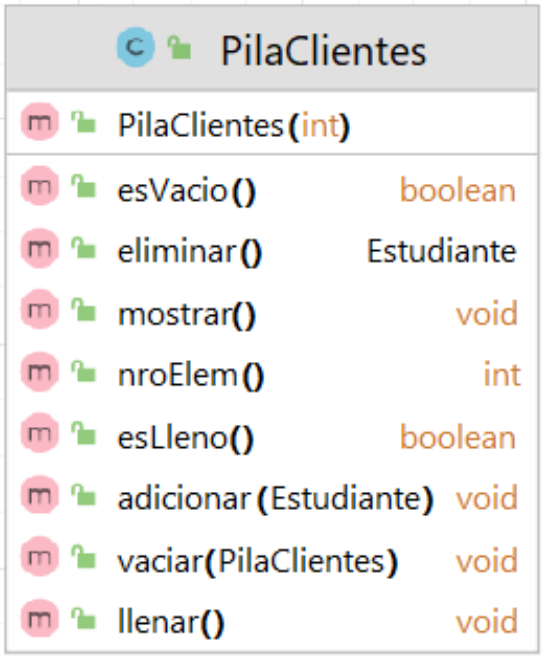

Estudiante
Estudiante(String, String, int, String, String)
Apellidos String
Semestre String
Genero String
Nombres String
Edad int
mostrarEstudiante() void
Edad int
Nombres String
Genero String
Apellidos String
Semestre String


ColaEstudiantes
ColaEstudiantes(int)
nroElem() int
adicionar(Estudiante) void
esVacia() boolean
esLlena() boolean
vaciar(ColaEstudiantes) void
mostrar() void
eliminar() Estudiante


Libro
Libro(String, String, int, double, String)
autor String
nroPag int
titulo String
codigoBook double
categoria String
mostrarLibro() void
autor String
codigoBook double
titulo String
nroPag int
categoria String


Main
Main()
main(String[]) void





Implementación

https://miro.com/welcomeonboard/VFA0em1aTXpoMkxCZ044RTRaR1hHVGdMb29YTHBhWEUxelRtZGFLZ3dCWDFqWndEekxNUVJxRFJmQkxYNERqZnwzMDc0NDU3MzY0MTU1NDI1MjcyfDI=?share_link_id=980919738909

Código JAVA, de todo el sistema.

CARRERA:

```
public class Carrera {
  private String nombreCarrera;
  private int noEstudiantes;
  private Estudiante[] estudiantes;

  public Carrera(String nombreCarrera, int noEstudiantes, Estudiante[]
estudiantes) {
      this.nombreCarrera = nombreCarrera;
      this.noEstudiantes = noEstudiantes;
      this.estudiantes = estudiantes;
  }
  public Carrera () {}

  public String getNombreCarrera() {
      return nombreCarrera;
  }
  public int getNoEstudiantes() {
      return noEstudiantes;
  }
  public Estudiante[] getEstudiantes() {
      return estudiantes;
  }
  public void setNombreCarrera(String nombreCarrera) {
      this.nombreCarrera = nombreCarrera;
  }
  public void setNoEstudiantes(int noEstudiantes) {
```

```
        this.noEstudiantes = noEstudiantes;
    }
    public void setEstudiantes(Estudiante[] estudiantes) {
        this.estudiantes = estudiantes;
    }
    public void mostrarDepartamento() {
        System.out.println("\nMOSTRANDO DATOS DE LA CARRERA");
        System.out.println("Nombre de la carrera: " +
this.getNombreCarrera());
        System.out.println("Nro de estudiantes: " + this.getNoEstudiantes());
        for (int i=0; i < this.noEstudiantes; i++) {
            this.getEstudiantes()[i].mostrarEstudiante();
        }
    }
}
```

COLAESTUDIANTE

```
public class ColaEstudiantes {
  private int max;
  private int fin;
  private int ini;
  private Estudiante[] Clientes;
  public ColaEstudiantes(int max) {
      this.max = max;
      this.Clientes  = new Estudiante[this.max + 1];
      this.ini = 0;
      this.fin = 0;
  }
  public boolean esVacia() {
```

```
    if (this.ini == 0 & this.fin == 0) {
        return true;
    } else {
        return false;
    }
}
public boolean esLlena() {
    if (this.fin == this.max) {
        return true;
    } else {
        return false;
    }
}
public int nroElem() {
    return fin - ini;
}

public void adicionar(Estudiante NuevoCliente) {
    if (!esLlena()) {
        fin++;
        Clientes[fin] = NuevoCliente;
    } else {
        System.out.println("Cola de numeros llena");
    }
}
public Estudiante eliminar() {
    Estudiante elementoEliminado = null;
    if (!esVacia()) {
        this.ini++;
```

```
            elementoEliminado = this.Clientes[ini];
            if (ini == fin) {
                ini = 0;
                fin = 0;
            }
            return elementoEliminado;
        } else {
            System.out.println("Cola de numeros vacia");
        }
        return elementoEliminado;
    }
    public void mostrar () {
        Estudiante elem = null;
        if (esVacia())
            System.out.println("Cola Vacia");
        else {
            System.out.println("\nDatos de la Cola de clientes");
            ColaEstudiantes aux = new ColaEstudiantes(fin);
            while (!esVacia()) {
                elem = this.eliminar();
                aux.adicionar (elem);
                elem.mostrarEstudiante();
            }
            vaciar(aux);
        }
    }
    public void vaciar (ColaEstudiantes cola) {
        while (!cola.esVacia())
            adicionar(cola.eliminar());
```

```
    }
}
```

ESTUDIANTE

```
  public class Estudiante {
    private String Nombres;
    private String Apellidos;
    private int Edad;
    private String Semestre;
    private String Genero;

    public Estudiante(String Nombres, String Apellidos, int Edad, String
Semestre, String Genero) {
        this.Nombres = Nombres;
        this.Apellidos = Apellidos;
        this.Edad = Edad;
        this.Semestre = Semestre;
        this.Genero = Genero;
    }
    public String getNombres() {
        return Nombres;
    }
    public String getApellidos() {
        return Apellidos;
    }
    public int getEdad() {
        return Edad;
    }
    public String getSemestre() {
```

```
    return Semestre;
}
public String getGenero() {
    return Genero;
}
public void setNombres(String nombres) {
    Nombres = nombres;
}
public void setApellidos(String apellidos) {
    Apellidos = apellidos;
}
public void setEdad(int edad) {
    Edad = edad;
}
public void setSemestre(String semestre) {
    Semestre = semestre;
}
public void setGenero(String genero) {
    Genero = genero;
}
public void mostrarEstudiante() {
    System.out.println("\nMostrando datos del estudiante");
    System.out.println("Nombre: " + this.getNombres());
    System.out.println("Apellidos: " + this.getApellidos());
    System.out.println("Edad: " + this.getEdad());
    System.out.println("Semestre: " + this.getSemestre());
    System.out.println("Genero: " + this.getGenero());
    System.out.println("\n");
```

```
    }
}
```

LIBRO

```
  public class Libro {
    private String titulo;
    private String autor;
    private int nroPag;
    private double codigoBook;
    private String categoria;

    public Libro(String titulo, String autor, int nroPag, double codigoBook,
String categoria) {
        this.titulo = titulo;
        this.autor = autor;
        this.nroPag = nroPag;
        this.codigoBook = codigoBook;
        this.categoria = categoria;
    }

    public String getTitulo() {
        return titulo;
    }

    public String getAutor() {
        return autor;
    }
    public int getNroPag() {
        return nroPag;
```

```
}
public double getCodigoBook() {
    return codigoBook;
}
public String getCategoria() {
    return categoria;
}
public void setTitulo(String titulo) {
    this.titulo = titulo;
}
public void setAutor(String autor) {
    this.autor = autor;
}
public void setNroPag(int nroPag) {
    this.nroPag = nroPag;
}
public void setCodigoBook(double codigoBook) {
    this.codigoBook = codigoBook;
}
public void setCategoria(String categoria) {
    this.categoria = categoria;
}
public void mostrarLibro(){
    System.out.println("\nMostrando libro");
    System.out.println("Titulo: " + this.getTitulo());
    System.out.println("Autor: " + this.getAutor());
    System.out.println("Nro de paginas: " + this.getNroPag());
    System.out.println("Codigo: " + this.getCodigoBook());
    System.out.println("Categoria: " + this.getCategoria());
```

```
    }
}
```

PILACLIENTES

```
  public class PilaClientes {
    private int max;
    private int tope;
    private Estudiante[] Clientes;
    public PilaClientes(int max) {
        this.tope = 0;
        this.max = max;
        this.Clientes = new Estudiante[this.max + 1];
    }
    public boolean esVacio () {
        if (tope == 0) {
            return true;
        } else {
            return false;
        }
    }
    public boolean esLleno () {
        if (tope == max) {
            return true;
        } else {
            return false;
        }
    }
    public int nroElem () {
        return this.tope;
```

```
}
public void adicionar (Estudiante nuevoCliente) {
    if (!this.esLleno()) {
        this.tope = this.tope + 1;
        this.Clientes[this.tope] = nuevoCliente;
    } else {
        System.out.println("La pila de numeros está llena");
    }
}
public Estudiante eliminar () {
    Estudiante elementoEliminado = null;
    if (!this.esVacio()) {
        elementoEliminado = (this.Clientes[this.tope]);
        this.tope = this.tope - 1;
    } else {
        System.out.println("La pila de libros está vacia");
    }
    return elementoEliminado;
}
public void llenar () {
}
public void mostrar () {
    Estudiante elem = null;
    if (esVacio())
        System.out.println("Pila Vacia");
    else {
        System.out.println("\nDatos de la Pila de clientes");
        PilaClientes aux = new PilaClientes(this.max);
        while (!esVacio()) {
```

```
                elem = this.eliminar();
                aux.adicionar (elem);
                elem.mostrarEstudiante();
            }
            vaciar(aux);
        }
    }
    public void vaciar (PilaClientes pila) {
        while (!pila.esVacio())
            adicionar(pila.eliminar());
    }
}
```

PILADELIBROS

```
  public class PilaDeLibros {
    private int max;
    private int tope;
    private Libro[] Libros;
    public PilaDeLibros(int max) {
        this.tope = 0;
        this.max = max;
        this.Libros = new Libro[this.max + 1];
    }
    public boolean esVacio () {
        if (tope == 0) {
            return true;
        } else {
            return false;
        }
```

```
}
public boolean esLleno () {
    if (tope == max) {
        return true;
    } else {
        return false;
    }
}
public int nroElem () {
    return this.tope;
}
public void adicionar (Libro nuevoLibro) {
    if (!this.esLleno()) {
        this.tope = this.tope + 1;
        this.Libros[this.tope] = nuevoLibro;
    } else {
        System.out.println("La pila de libros está llena");
    }
}
public Libro eliminar () {
    Libro elementoEliminado = null;
    if (!this.esVacio()) {
        elementoEliminado = (this.Libros[this.tope]);
        this.tope = this.tope - 1;
    } else {
        System.out.println("La pila de libros está vacia");
    }
    return elementoEliminado;
}
```

```
    public void llenar () {
    }
    public void mostrar () {
        Libro elem = null;
        if (esVacio())
            System.out.println("Pila Vacia");
        else {
            System.out.println(" Datos de la Pila de libros");
            PilaDeLibros aux = new PilaDeLibros(this.max);
            while (!esVacio()) {
                elem = this.eliminar();
                aux.adicionar (elem);
                elem.mostrarLibro();
            }
            vaciar(aux);
        }
    }
    public void vaciar (PilaDeLibros pila) {
        while (!pila.esVacio()) {
            adicionar(pila.eliminar());
        }
    }
}
```

MAIN

```
package Proyecto_X;

import java.util.Collections;
import java.util.Random;
```

```
import java.util.Scanner;


public class Main {
    public static void main(String[] args) {
    //Colas de estudiantes
        Estudiante cli1 = new Estudiante("Josias", "Leon", 33, "Tercero",
"Masculino");
        Estudiante cli2 = new Estudiante("Adolf", "Hitler", 56, "Quinto",
"Masculino");
        Estudiante cli3 = new Estudiante("Alejandra", "Maine", 19, "Segundo",
"Femenino");
        Estudiante cli4 = new Estudiante("Josef", "Stalin", 74, "Quinto",
"Masculino");
        Estudiante cli5 = new Estudiante("Saul", "Saulero", 22, "Primero",
"Masculino");


        ColaEstudiantes Cola = new ColaEstudiantes(100);
        Cola.adicionar(cli1);
        Cola.adicionar(cli2);
        Cola.adicionar(cli3);
        Cola.adicionar(cli4);
        Cola.adicionar(cli5);


        Estudiante cli6 = new Estudiante("Natalia", "Poklonskaya", 22,
"Segundo", "Femenino");
        Estudiante cli7 = new Estudiante("Adrian", "Fernandez", 23, "Quinto",
"Masculino");
        Estudiante cli8 = new Estudiante("Leo", "Gallardo", 24, "Septimo",
```

```
"Masculino");
        Estudiante cli9 = new Estudiante("Len", "Bluxen", 20, "Cuarto",
"Masculino");
        Estudiante cli10 = new Estudiante("Alice", "Wright", 19, "Segundo",
"Femenino");

        ColaEstudiantes Cola2 = new ColaEstudiantes(100);
        Cola2.adicionar(cli6);
        Cola2.adicionar(cli7);
        Cola2.adicionar(cli8);
        Cola2.adicionar(cli9);
        Cola2.adicionar(cli10);

        Estudiante cli11 = new Estudiante("Ilia", "Sarzo", 20, "Tercero",
"Femenino");
        Estudiante cli12 = new Estudiante("Andres", "Quiroga", 172,
"Tercero", "Masculino");
        Estudiante cli13 = new Estudiante("Marco", "Calle", 19, "Cuarto",
"Masculino");
        Estudiante cli14 = new Estudiante("Iris", "Velasco", 5, "Octavo",
"Femenino");
        Estudiante cli15 = new Estudiante("Cristian", "Machicado", 42,
"Quinto", "Masculino");

        ColaEstudiantes Cola3 = new ColaEstudiantes(100);
        Cola3.adicionar(cli11);
        Cola3.adicionar(cli12);
        Cola3.adicionar(cli13);
        Cola3.adicionar(cli14);
```

```
        Cola3.adicionar(cli15);


        Estudiante cli16 = new Estudiante("Pepe", "Mamani", 25, "Cuarto",
"Masculino");
        Estudiante cli17 = new Estudiante("Armando", "Casas", 17, "Tercero",
"Masculino");
        Estudiante cli18 = new Estudiante("Domingo", "Calisaya", 21, "Sexto",
"Masculino");
        Estudiante cli19 = new Estudiante("Roberta", "Siñani", 25, "Septimo",
"Femenino");
        Estudiante cli20 = new Estudiante("Emma", "Stone", 22, "Tercero",
"Femenino");


        ColaEstudiantes Cola4 = new ColaEstudiantes(100);
        Cola4.adicionar(cli16);
        Cola4.adicionar(cli17);
        Cola4.adicionar(cli18);
        Cola4.adicionar(cli19);
        Cola4.adicionar(cli20);


        //Pilas de Libros
        Libro li1 = new Libro("Boulevard","Flor",300,131,"Novela");
        Libro li2 = new Libro("la fabula de
Esopo","anonimo",150,512,"Cuentos");
        Libro li3 = new Libro("Geografia","Santillan",400,842,"Educativo");
        Libro li4 = new Libro("Guerra y paz","Leon
toistoi",244,312,"Novela");
        Libro li5 = new Libro("Don Quijote","Miguel
Servantes",300,122,"Novela");
```

```
        PilaDeLibros estante1 = new PilaDeLibros(100);
        estante1.adicionar(li1);
        estante1.adicionar(li2);
        estante1.adicionar(li3);
        estante1.adicionar(li4);
        estante1.adicionar(li5);

        Libro li6 = new Libro("Kamasutra","Bráhmana
Vatsiaiana",200,2469,"Cuentos");
        Libro li7 = new Libro("Kamehasutra","Pandora's
Box",150,443,"Cuentos");
        Libro li8 = new Libro("La caja de Pandora","Pablo
Riba",164,746,"Educativo");
        Libro li9 = new Libro("Luna de Plutón","Angel David
Revilla",472,464,"Novela");
        Libro li10 = new Libro("Anatomia Humana","Frank
H",300,263,"Educativo");

        PilaDeLibros estante2 = new PilaDeLibros(100);
        estante2.adicionar(li6);
        estante2.adicionar(li7);
        estante2.adicionar(li8);
        estante2.adicionar(li9);
        estante2.adicionar(li10);
        Libro li11 = new Libro("El gato negro","Edgar Allan
Poe",288,659,"Novela");
        Libro li12 = new Libro("It","Stephen King",200,575,"Terror");
        Libro li13 = new Libro("Romeo y
Julieta","Shakespeare",200,646,"Novela");
```

```
        Libro li14 = new Libro("Free Sex","Carlos Cuautemoc
Sanchez",152,652,"Educativo");
        Libro li15 = new Libro("Sangre de Campeon","Carlos Cuautemoc
Sanchez",200,444,"Novela");

        PilaDeLibros estante3 = new PilaDeLibros(100);
        estante3.adicionar(li11);
        estante3.adicionar(li12);
        estante3.adicionar(li13);
        estante3.adicionar(li14);
        estante3.adicionar(li15);

        Libro li16 = new Libro("Tempestad en la cordillera","Walter Guevara
Arce",100,564,"Novela");
        Libro li17 = new Libro("Perdon a la lluvia","Sara
Buho",100,534,"Poesia");
        Libro li18 = new Libro("Biblia","varios",600,7777777,"Historico");
        Libro li19 = new Libro("Hamlet","Shakespeare",300,746,"Novela");
        Libro li20 = new Libro("Energia en
Bolivia","Anonimo",10,563,"Informativo");

        PilaDeLibros estante4 = new PilaDeLibros(100);
        estante4.adicionar(li16);
        estante4.adicionar(li17);
        estante4.adicionar(li18);
        estante4.adicionar(li19);
        estante4.adicionar(li20);

        //kEsimo
```

```
        //CambiarSentido
        //anadirLibro(estante3);
        //cambiarColaEstudiantes(Cola,Cola2,"Quinto","Segundo");
        organizarLibros(estante1,estante2,"Novela","Educativo");

    }
    public static void kEsimoCliente(PilaDeLibros pila, int valorTope){
        PilaDeLibros aux = new PilaDeLibros(10);
        Libro valor = null;

        while (!pila.esVacio()){
            if (pila.nroElem() != valorTope){
                aux.adicionar(pila.eliminar());
            }
            else {
                valor = pila.eliminar();
            }
        }
        pila.vaciar(aux);
        pila.adicionar(valor);
        pila.mostrar();
    }

    public static void CambiarSentido(PilaDeLibros pila) {
        PilaDeLibros aux = new PilaDeLibros(10);
        Libro ultimolibroEliminado = pila.eliminar();
        Libro libroEliminado = null;
        Libro primerLibroEliminado = null;
        while (!pila.esVacio()) {
```

```
            aux.adicionar(pila.eliminar());
        }
        primerLibroEliminado = aux.eliminar();
        aux.adicionar(ultimolibroEliminado);
        pila.vaciar(aux);
        pila.adicionar(primerLibroEliminado);
        pila.mostrar();
    }
    public static void  anadirLibro(PilaDeLibros pila){
        Scanner leer = new Scanner(System.in);
        System.out.println("INGRESE EL NUEVO LIBRO");
        System.out.println("ingrese el titulo del libro: ");
        String titulo = leer.nextLine();
        System.out.println("ingrese el autor del libro: ");
        String autor = leer.nextLine();
        System.out.println("ingrese el numero de paginas del libro: ");
        int nroPag = Integer.parseInt(leer.next());
        System.out.println("ingrese el codigo del libro: ");
        int CodigoLibro = Integer.parseInt(leer.next());
        System.out.println("ingrese la categoria del libro: ");
        String categoria = leer.next();

        Libro nuevolibro = new
Libro(titulo,autor,nroPag,CodigoLibro,categoria);
        pila.adicionar(nuevolibro);
        pila.mostrar();
    }
```

//Este método sirve para ordenar de mejor forma a los estudiantes de las colas, al ordenarlos

por semestre

```
    public static void cambiarColaEstudiantes(ColaEstudiantes cola1,
ColaEstudiantes cola2,String Semestre1,String Semestre2) {
        int nroElemColaA = cola1.nroElem();
        int nroElemColaB = cola2.nroElem();
        ColaEstudiantes aux = new ColaEstudiantes(100);
        ColaEstudiantes aux2 = new ColaEstudiantes(100);

        Estudiante valorEliminado = null;
        for (int i = 1; i <= nroElemColaA; i++) {
            valorEliminado = cola1.eliminar();
            if (valorEliminado.getSemestre().equals(Semestre1)) {
                cola2.adicionar(valorEliminado);
            } else {
                aux.adicionar(valorEliminado);
            }
        }
        for (int i = 1; i <= nroElemColaB; i++) {
            valorEliminado = cola2.eliminar();
            if (valorEliminado.getSemestre().equals(Semestre2)) {
                cola1.adicionar(valorEliminado);
            } else {
                aux2.adicionar(valorEliminado);
            }
        }
        cola1.vaciar(aux);
        cola2.vaciar(aux2);
        cola1.mostrar();
```

```
        cola2.mostrar();
    }

    public static void organizarLibros(PilaDeLibros pila1, PilaDeLibros
pila2, String Categoria1,String Categoria2) {
        int nrodeLibro1 = pila1.nroElem();
        int nrodeLibro2 = pila2.nroElem();
        PilaDeLibros aux = new PilaDeLibros(100);
        PilaDeLibros aux2 = new PilaDeLibros(100);

        Libro valorEliminado = null;
        for (int i = 1; i <= nrodeLibro1; i++) {
            valorEliminado = pila1.eliminar();
            if (valorEliminado.getCategoria().equals(Categoria2)) {
                pila2.adicionar(valorEliminado);
            } else {
                aux.adicionar(valorEliminado);
            }
        }
        for (int i = 1; i <= nrodeLibro2; i++) {
            valorEliminado = pila2.eliminar();
            if (valorEliminado.getCategoria().equals(Categoria1)) {
                pila1.adicionar(valorEliminado);
            } else {
                aux2.adicionar(valorEliminado);
            }
        }
        pila1.vaciar(aux);
        pila2.vaciar(aux2);
```

```
        pila1.mostrar();
        pila2.mostrar();
    }
```

//cambiar datos de un estudiante de una cola a otra

```
    public static void cambiarDatos ( ColaEstudiantes cola1, ColaEstudiantes
cola2 ) {
        Estudiante estudiante = cola1.eliminar();
        cola2.adicionar(estudiante);
        cola1.mostrar();
        cola2.mostrar();
    }
```

//crear un método que elimine un libro por su posición

//recibe como parámetro el estante y la posición del libro a eliminar

```
    public static void eliminarLibro ( PilaDeLibros estante, int posicion ) {
        PilaDeLibros aux = new PilaDeLibros(10);
        Libro libro = null;
        while (!estante.esVacio()) {
            libro = estante.eliminar();
            if (estante.nroElem() != posicion) {
                aux.adicionar(libro);
            }
        }
        estante.vaciar(aux);
        estante.mostrar();
```

```
}
```

//crear un método que elija un libro al azar de un estante

//recibe como parámetro el estante y retorna el libro elegido

```
public static Libro libroAlAzar ( PilaDeLibros estante ) {
    Random random = new Random();
    int posicion = random.nextInt(estante.nroElem());
    PilaDeLibros aux = new PilaDeLibros(10);
    Libro libro = null;
    while (!estante.esVacio()) {
        libro = estante.eliminar();
        if (estante.nroElem() != posicion) {
            aux.adicionar(libro);
        }
    }
    estante.vaciar(aux);
    return libro;
}
```

//metodo que elimina el ultimo estudiante de la cola

```
public static void eliminarUltimoEstudiante ( ColaEstudiantes cola ) {
    ColaEstudiantes aux = new ColaEstudiantes(10);
    Estudiante estudiante = null;
    while (!cola.esVacia()) {
        estudiante = cola.eliminar();
        if (!cola.esVacia()) {
            aux.adicionar(estudiante);
        }
```

```
        }
        cola.vaciar(aux);
        cola.mostrar();
    }
```

//crear un método que busque libros por su categoría

//y muestre el número de libros que hay en el estante

```
    public static void buscarLibrosPorCategoria ( PilaDeLibros estante,
String categoria ) {
        PilaDeLibros aux = new PilaDeLibros(10);
        Libro libro = null;
        int contador = 0;
        while (!estante.esVacio()) {
            libro = estante.eliminar();
            if (libro.getCategoria().equals(categoria)) {
                contador++;
            }
            aux.adicionar(libro);
        }
        estante.vaciar(aux);
        System.out.println("Hay " + contador + " libros de la categoria " +
categoria);
    }
```

//crear un metodo que busque libros por su autor

//y muestre el numero de libros que hay en el estante

```
    public static void buscarLibrosPorAutor ( PilaDeLibros estante, String
autor ) {
```

```
        PilaDeLibros aux = new PilaDeLibros(10);
        Libro libro = null;
        int contador = 0;
        while (!estante.esVacio()) {
            libro = estante.eliminar();
            if (libro.getAutor().equals(autor)) {
                contador++;
            }
            aux.adicionar(libro);
        }
        estante.vaciar(aux);
        System.out.println("Hay " + contador + " libros del autor " + autor);
    }
}
```

Usabilidad

//Este método sirve para mover de un libro de su posición actual al inicio

// de la pila

```
 Datos de la Pila de libros

Mostrando libro
Titulo: Guerra y paz
Autor: Leon toistoi
Nro de paginas: 244
Precio: 312
Categoria: Novela

Mostrando libro
Titulo: Don Quijote
Autor: Miguel Servantes
Nro de paginas: 300
Precio: 122
Categoria: Novela

Mostrando libro
Titulo: Geografia
Autor: Santillan
Nro de paginas: 400
Precio: 842
Categoria: Educativo

Mostrando libro
Titulo: la fabula de Esopo
Autor: anonimo
Nro de paginas: 150
Precio: 512
Categoria: Cuentos

Mostrando libro
Titulo: Boulevard
Autor: Flor
Nro de paginas: 300
Precio: 131
Categoria: Novela
```

// Este método cambia sentido el último libro se va al inicio

// y el primero libro se va al final

```
 Datos de la Pila de libros

Mostrando libro
Titulo: Tempestad en la cordillera
Autor: Walter Guevara Arce
Nro de paginas: 100
Precio: 564
Categoria: Novela

Mostrando libro
Titulo: Hamlet
Autor: Shakespeare
Nro de paginas: 300
Precio: 746
Categoria: Novela

Mostrando libro
Titulo: Biblia
Autor: varios
Nro de paginas: 600
Precio: 7777777
Categoria: Historico

Mostrando libro
Titulo: Perdon a la lluvia
Autor: Sara Buho
Nro de paginas: 100
Precio: 534
Categoria: Poesia

Mostrando libro
Titulo: Energia en Bolivia
Autor: Anonimo
Nro de paginas: 10
Precio: 563
Categoria: Informativo
```

// Este método sirve para añadir un nuevo libro a la pila seleccionada

```
INGRESE EL NUEVO LIBRO
ingrese el titulo del libro:
ultimos recuerdos del mundo
ingrese el autor del libro:
josias
ingrese el numero de paginas del libro:
200
ingrese el codigo del libro:
1435464
ingrese la categoria del libro:
Novela
 Datos de la Pila de libros

Mostrando libro
Titulo: ultimos recuerdos del mundo
Autor: josias
Nro de paginas: 200
Precio: 1435464
Categoria: Novela

Mostrando libro
Titulo: Sangre de Campeon
Autor: Carlos Cuautemoc Sanchez
Nro de paginas: 200
Precio: 444
Categoria: Novela

Mostrando libro
Titulo: Free Sex
Autor: Carlos Cuautemoc Sanchez
Nro de paginas: 152
Precio: 652
Categoria: Educativo

Mostrando libro
Titulo: Romeo y Julieta
Autor: Shakespeare
Nro de paginas: 200
Precio: 646
Categoria: Novela

Mostrando libro
Titulo: It
Autor: Stephen King
Nro de paginas: 200
Precio: 575
Categoria: Terror

Mostrando libro
Titulo: El gato negro
Autor: Edgar Allan Poe
Nro de paginas: 288
Precio: 659
Categoria: Novela

Process finished with exit code 0
```

//Este método sirve para ordenar de mejor forma a los estudiantes de las colas, al ordenarlos por semestre

```
Datos de la Cola de clientes

Mostrando datos del estudiante
Nombre: Natalia
Apellidos: Poklonskaya
Edad: 22
Semestre: Segundo
Genero: Femenino

Mostrando datos del estudiante
Nombre: Alice
Apellidos: Wright
Edad: 19
Semestre: Segundo
Genero: Femenino

Mostrando datos del estudiante
Nombre: Josias
Apellidos: Leon
Edad: 33
Semestre: Tercero
Genero: Masculino

Mostrando datos del estudiante
Nombre: Alejandra
Apellidos: Maine
Edad: 19
Semestre: Segundo
Genero: Femenino

Mostrando datos del estudiante
Nombre: Saul
Apellidos: Saulero
Edad: 22
Semestre: Primero
Genero: Masculino

Datos de la Cola de clientes

Mostrando datos del estudiante
Nombre: Adolf
Apellidos: Hitler
Edad: 56
Semestre: Quinto
Genero: Masculino

Mostrando datos del estudiante
Nombre: Josef
Apellidos: Stalin
Edad: 74
Semestre: Quinto
Genero: Masculino

Mostrando datos del estudiante
Nombre: Adrian
Apellidos: Fernandez
Edad: 23
Semestre: Quinto
Genero: Masculino
```

// Este método sirve para poder organizar Libros entre dos estantes moviendo de un estante al otro y viceversa por categoría

```
Mostrando libro
Titulo: Don Quijote
Autor: Miguel Servantes
Nro de paginas: 300
Precio: 122
Categoria: Novela

Mostrando libro
Titulo: Guerra y paz
Autor: Leon toistoi
Nro de paginas: 244
Precio: 312
Categoria: Novela

Mostrando libro
Titulo: la fabula de Esopo
Autor: anonimo
Nro de paginas: 150
Precio: 512
Categoria: Cuentos

Mostrando libro
Titulo: Boulevard
Autor: Flor
Nro de paginas: 300
Precio: 131
Categoria: Novela

Mostrando libro
Titulo: Luna de Plut◆n
Autor: Angel David Revilla
Nro de paginas: 472
Precio: 464
Categoria: Novela
 Datos de la Pila de libros

Mostrando libro
Titulo: Geografia
Autor: Santillan
Nro de paginas: 400
Precio: 842
Categoria: Educativo

Mostrando libro
Titulo: Anatomia Humana
Autor: Frank H
Nro de paginas: 300
Precio: 263
Categoria: Educativo

Mostrando libro
Titulo: La caja de Pandora
Autor: Pablo Riba
Nro de paginas: 164
Precio: 746
Categoria: Educativo

Mostrando libro
Titulo: Kamehasutra
Autor: Pandora's Box
Nro de paginas: 150
Precio: 443
Categoria: Cuentos

Mostrando libro
Titulo: Kamasutra
Autor: Br◆hmana Vatsiaiana
Nro de paginas: 200
Precio: 2469
Categoria: Cuentos

Process finished with exit code 0
```

//Cambiar datos de un estudiante de una cola a otra

```
Datos de la Cola de clientes

Mostrando datos del estudiante
Nombre: Adolf
Apellidos: Hitler
Edad: 56
Semestre: Quinto
Genero: Masculino
```

```
Datos de la Cola de clientes

Mostrando datos del estudiante
Nombre: Natalia
Apellidos: Poklonskaya
Edad: 22
Semestre: Segundo
Genero: Femenino
```

```
Mostrando datos del estudiante
Nombre: Alejandra
Apellidos: Maine
Edad: 19
Semestre: Segundo
Genero: Femenino
```

```
Mostrando datos del estudiante
Nombre: Adrian
Apellidos: Fernandez
Edad: 23
Semestre: Quinto
Genero: Masculino
```

```
Mostrando datos del estudiante
Nombre: Josef
Apellidos: Stalin
Edad: 74
Semestre: Quinto
Genero: Masculino
```

```
Mostrando datos del estudiante
Nombre: Leo
Apellidos: Gallardo
Edad: 24
Semestre: Septimo
Genero: Masculino
```

```
Mostrando datos del estudiante
Nombre: Saul
Apellidos: Saulero
Edad: 22
Semestre: Primero
Genero: Masculino
```

```
Mostrando datos del estudiante
Nombre: Len
Apellidos: Bluxen
Edad: 20
Semestre: Cuarto
Genero: Masculino
```

```
Mostrando datos del estudiante
Nombre: Alice
Apellidos: Wright
Edad: 19
Semestre: Segundo
Genero: Femenino


Mostrando datos del estudiante
Nombre: Josias
Apellidos: Leon
Edad: 33
Semestre: Tercero
Genero: Masculino


Process finished with exit code 0
```

//crear un método que elimine un libro por su posición

//recibe como parámetro el estante y la posición del libro a eliminar

```
 Datos de la Pila de libros

Mostrando libro
Titulo: Guerra y paz
Autor: Leon toistoi
Nro de paginas: 244
Precio: 312
Categoria: Novela

Mostrando libro
Titulo: Geografia
Autor: Santillan
Nro de paginas: 400
Precio: 842
Categoria: Educativo

Mostrando libro
Titulo: la fabula de Esopo
Autor: anonimo
Nro de paginas: 150
Precio: 512
Categoria: Cuentos

Mostrando libro
Titulo: Boulevard
Autor: Flor
Nro de paginas: 300
Precio: 131
Categoria: Novela

Process finished with exit code 0
```

//Método que elimina el ultimo estudiante de la cola

```
Datos de la Cola de clientes

Mostrando datos del estudiante
Nombre: Josias
Apellidos: Leon
Edad: 33
Semestre: Tercero
Genero: Masculino


Mostrando datos del estudiante
Nombre: Adolf
Apellidos: Hitler
Edad: 56
Semestre: Quinto
Genero: Masculino


Mostrando datos del estudiante
Nombre: Alejandra
Apellidos: Maine
Edad: 19
Semestre: Segundo
Genero: Femenino


Mostrando datos del estudiante
Nombre: Josef
Apellidos: Stalin
Edad: 74
Semestre: Quinto
Genero: Masculino
```

//crear un método que busque libros por su categoría

//y muestre el número de libros que hay en el estante

```
C:\Users\SARZO\.jdks\openjdk-18.0.2\bin\java.exe "
Hay 3 libros de la categoria Novela

Process finished with exit code 0
```

//crear un método que busque libros por su autor

//y muestre el número de libros que hay en el estante

```
C:\Users\SARZO\.jdks\openjdk-18.0.2\bin
Hay 0 libros del autor mario

Process finished with exit code 0
```

LINK DEL VIDEO

https://m.facebook.com/story.php?story_fbid=pfbid04bKsDYQTJjjmKf8e4EWiUow9KsUemFt3VpBZmGAX9TS5gJtwsZrVi1PZVaX1e8Gal&id=100082796172894&mibextid=Nif5oz

CAPITULO IV

4.1 Conclusiones

Luego de haber concluido con el proyecto de Estructura de Datos sobre el programa de Proyecto Biblioteca fueron muchos los esfuerzos y conocimientos adquiridos durante el semestre y que fueron plasmados en este proyecto.

Cada problema que nos establecimos fue realizado y solucionado hubo algunas dificultades, pero aun así se pudo cumplir con cada problema planteado.

Al realizar el proyecto nos dimos cuenta de varios conceptos importantes que no serán de mucha ayuda a lo largo de nuestra carrera.