Jessica Zepeda Baeza
Sistemas Operativos
Tarea 01

```java
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;

/**
 * Clase de Polifase.
 * @author Barreiro Valdez Alejandro, Piña Félix Emilio, Zepeda Baeza Jessica
 */
public class Polifase {
    private File f1, f2, f3, f4;
    int atributo;

    /**
     * Operaciones de la primera fase de Polifase.
     * @param s Scanner del archivo original.
     * @throws IOException
     */
    private void primeraFase(Scanner s) throws IOException {
        ArrayList<Alumno> lista_alumnos = new ArrayList<>(); //Lista de bloques de n de alumnos
        MergeSort m1 = new MergeSort(atributo); //MergeSort con el atributo a comparar.

        /*Lee el archivo hasta que no haya más líneas.*/
        while (s.hasNextLine()) {
            lista_alumnos.clear(); //Limpia la lista de las iteraciones anteriores.
            leerN(lista_alumnos, 2, s); //Lee y crea n alumnos en una lista.
            m1.sort(lista_alumnos, 0, lista_alumnos.size() - 1); //Ordena los bloques
            FileWriter myWriter1 = new FileWriter(f1, true);

            /*Escribe los bloques ordenados en el archivo 1.*/
            for (Alumno a : lista_alumnos) {
                myWriter1.write(a.toString());
            }
            myWriter1.write("|\n");
            myWriter1.close();

            /*Se repite el proceso para el archivo 2*/
            lista_alumnos.clear();
            leerN(lista_alumnos, 2, s);
            m1.sort(lista_alumnos, 0, lista_alumnos.size() - 1);
            FileWriter myWriter2 = new FileWriter(f2, true);

            for (Alumno a : lista_alumnos) {
                myWriter2.write(a.toString());
            }
            myWriter2.write("|\n");
            myWriter2.close();
        }
    }

    /**
     * Método para que si ya existe un archivo de iterciones anteriores, éste se borra.
     * Si no existe solo se crea.
     * @param fileName Nombre del archivo a crear.
     * @return Un nuevo archivo donde se sobreescribe.
     * @throws IOException
     */
    private File newFile(String fileName)throws IOException{
        File f = new File(fileName);
        if(f.createNewFile())
            return f;
        f.delete();
```

- es una llamada al sistema ya que realiza el servicio de "leer" e indicar si existe una entrada.

- llamadas al sistema que modifican archivos: realizan un servicio al escribir, crear, cerrar, borrar y leer archivos

```
        return new File(fileName);
    }

    /**
     * Menú donde se ingresa el nombre del archivo a leer.
     */
    public static void menu(){
        int i = 1;
        do{
            Scanner scan = new Scanner(System.in);
            System.out.print("Ingrese el nombre del archivo: ");
            String s = scan.next();
            try{
                File archivo = new File(s);
                Scanner scf = new Scanner(archivo);
                System.out.println("Elige una categoría para ordenar: ");
                System.out.println("1.Claves\n2.Nombres\n3.Apellidos");
                int eleccion = scan.nextInt();
                Polifase p1 = new Polifase(eleccion);
                p1.primeraFase(scf);
                p1.segundaFase();
                System.out.println("Terminado!\nIngresa (1) para repetir la ejecución y (0) para salir");
                i = scan.nextInt();
                scf.close();
            }catch(IOException ex){
                System.out.println(ex.getMessage());
            }
        }while(i!=0);
    }

}
```

es una llamada al sistema que realiza el servicio de mostrar una salida en pantalla

es una llamada al sistema ya que hace el servicio de leer caracteres ingresados por el usario