Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software.

Nombre de la Práctica Arreglos unidimensionales

Numero de Práctica 4

Nombre de la carrera Ingeniería de Software

Nombre de la materia Lab. Estructuras de Datos

Nombre del alumno Jesús Manuel Juárez Pasillas

Nombre del docente Aldonso Becerra Sánchez

Fecha: 25/08/2021

Práctica 4: Arreglos unidimensionales

Introducción:

Manejar información guardada en arreglos permite resolver una gran cantidad de problemas puestos por un programa, en cuanto a la información que se requiere mostrar, los datos que son importantes para un fácil manejo de la información. Si la información se almacena de forma ordenada nos permite acceder mucho más fácil a los datos, así como manipularlos.

Desarrollo:

Para el desarrollo de esta práctica se hará uso de la herencia, ya que para la creación de la clase ArregloOrden es necesario heredar de la clase ArregloDatos, como esta clase ya contiene métodos, algunos servirán plenamente para ArregloOrden pero algunos no por lo que se tendrán que sobrescribir.

Se agregaron 4 clases nuevas al proyecto una de las cuales es una prueba para probar el funcionamiento de lo agregado y otra es un enumerado el cual tiene 2 opciones (ASCENDENTE y DESCENDENTE).

ArregloOrden: En esta clase se encuentra dentro del paquete estructuraslineales y en ella se agregaron los siguientes métodos:

```
- Inserta al final de la lista un elemento proporcionado.
  public int agregar(Object elemento);

- Busca en el arreglo el elemento requerido.
  public Object buscar(Object elemento);

- Elimina el dato solicitado del arreglo.
  public Object eliminar(Object elemento);

- Modifica un elemento que se quiere.
  public boolean cambiar(Object elementoViejo, Object
  elementoNuevo, int numVeces);

- Modifica un elemento de la lista en base al índice.
  public boolean cambiar(int indice, Object elemento);

- Agrega los datos de la lista pasada como parámetro a la lista
  actual.
  public boolean agregarLista(Object listaDatos2);

- Invierte el orden de los elementos de la lista.
  public void invertir();

- Rellena dependiendo si es un número, letra o un objeto.
  public void rellenar(Object elemento);
```

```
- Desordena los elementos del arreglo y los introduce en un arreglo
  normal.
  public ArregloDatos arregloDesordenado();

- Indica si la lista pasada como parámetro es sublista de la lista
  actual.
  public boolean esSublista(Object listaDatos2);
- Cambia los datos en común de la lista actual y la listaDatos2 y
  lo cambia por el elemento que está en listaDatos2Nuevos.
  public boolean cambiarLista(ArregloDatos listaDatos2,ArregloDatos
  listaDatos2Nuevos);

- Deja en la lista actual solo los elementos que se encuentran en
  listaDatos2.
  public boolean retenerLista(ArregloDatos listaDatos2);

- Inserta en la posición indicada el elemento.
  public boolean insertar(int indice, Object elemento);

- Agrega los datos de la lista a la lista actual quitando los
  anteriores.
  public boolean copiarLista(ArregloDatos listaDatos2);
```

Herramientas: Esta clase se encuentra en el paquete utilerías/comunes, en ella se agregó el método de compararObjetos.

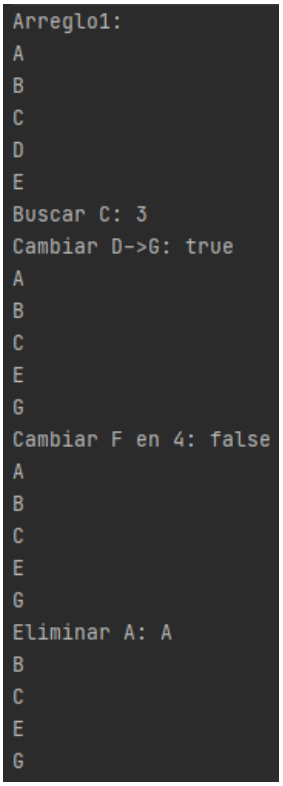
```
- Compara objetos, dependiendo si son números o algún otro tipo de
  objetos.
  public static int compararObjetos(Object objeto1, Object objeto2)
```

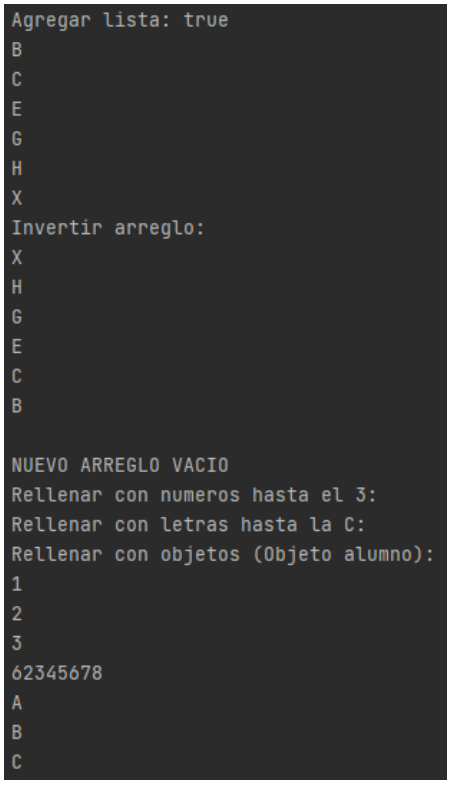
ArregloDatos: en esta clase se agregaron 6 métodos nuevos:

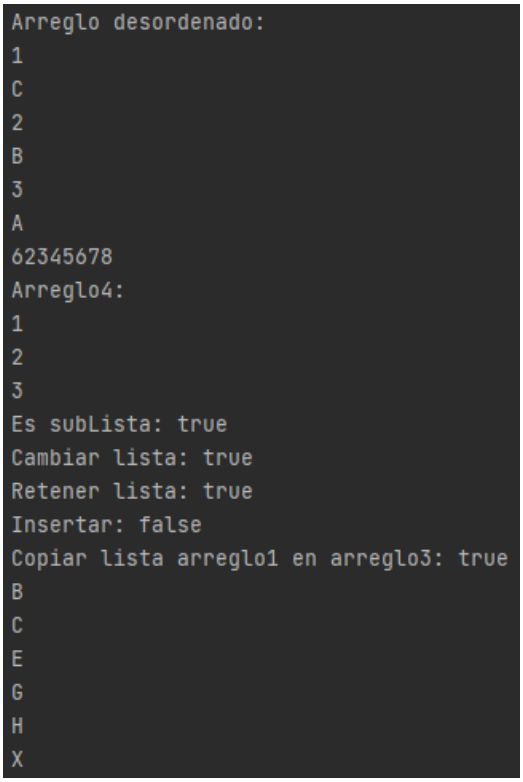
```
- Rellena dependiendo si es un número, letra o un objeto.
  public void rellenar(Object elemento);
- Indica si la lista pasada como parámetro es sublista de la lista
  actual.
  public boolean esSublista(Object listaDatos2);
- Cambia los datos en común de la lista actual y la listaDatos2 y
  lo cambia por el elemento que está en listaDatos2Nuevos.
  public boolean cambiarLista(ArregloDatos listaDatos2,ArregloDatos
  listaDatos2Nuevos);

- Deja en la lista actual solo los elementos que se encuentran en
  listaDatos2.
  public boolean retenerLista(ArregloDatos listaDatos2);

- Inserta en la posición indicada el elemento.
  public boolean insertar(int indice, Object elemento);

- Agrega los datos de la lista a la lista actual quitando los
  anteriores.
  public boolean copiarLista(ArregloDatos listaDatos2);
```

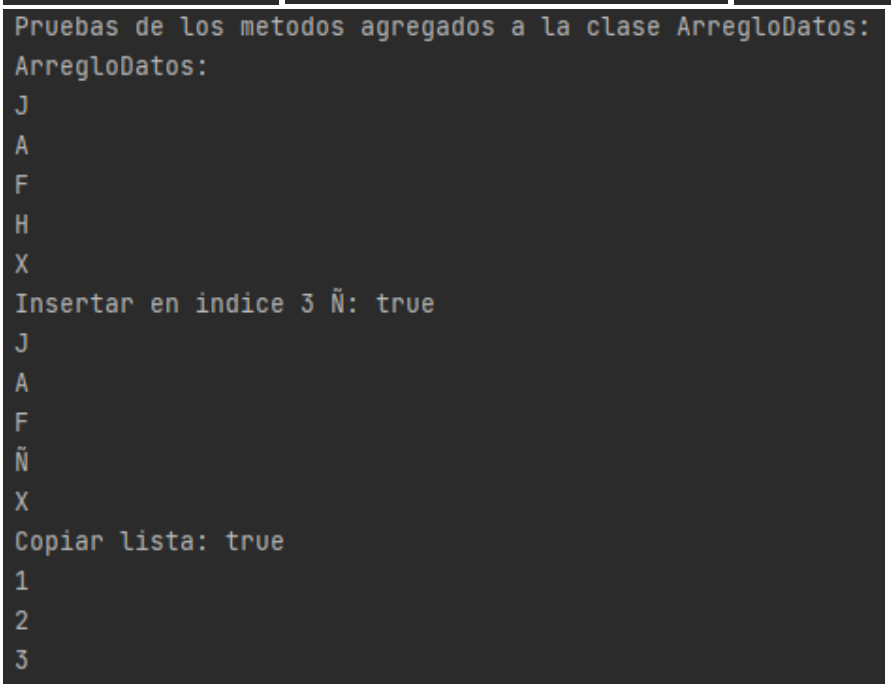
Nota: Toda la documentación esta agregada en la carpeta “doc” dentro de la carpeta del proyecto (“edylab_2021_3/doc”).

Capturas del programa funcionando:



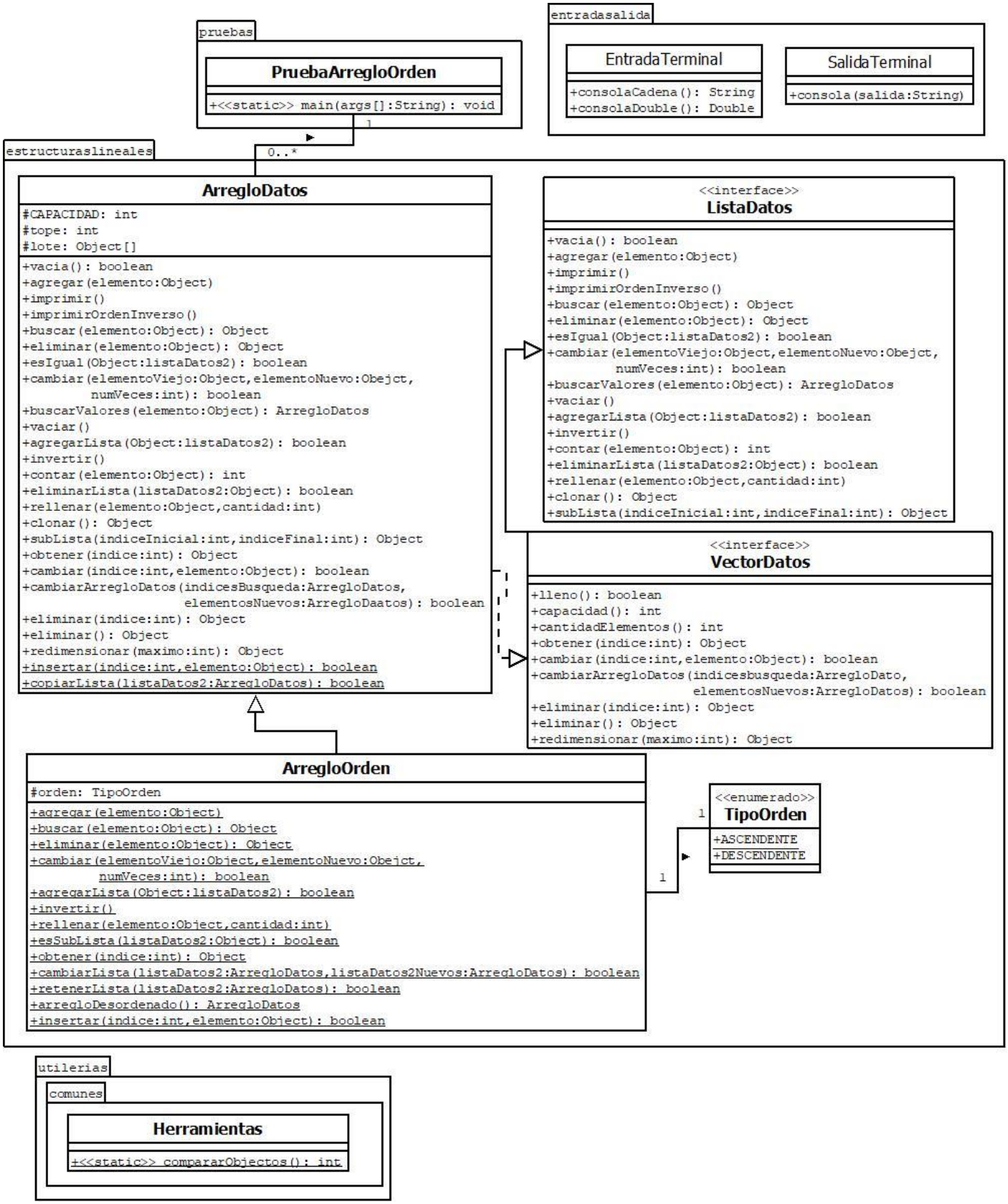






Código agregado:

Las clases ArregloOrden, Herramientas y PruebaArregloOrden son nuevas, además todos los metodos subrayados son nuevos.



Pre-evaluación:

Pre-Evaluación para prácticas de Laboratorio de Estructuras de Datos	PRE-EVALUACIÓN DEL ALUMNO
CUMPLE CON LA FUNCIONALIDAD SOLICITADA.	Sí
DISPONE DE CÓDIGO AUTO-DOCUMENTADO.	Sí
DISPONE DE CÓDIGO DOCUMENTADO A NIVEL DE CLASE Y MÉTODO.	Sí
DISPONE DE INDENTACIÓN CORRECTA.	Sí
CUMPLE LA POO.	Sí
DISPONE DE UNA FORMA FÁCIL DE UTILIZAR EL PROGRAMA PARA EL USUARIO.	Sí
DISPONE DE UN REPORTE CON FORMATO IDC.	Sí
LA INFORMACIÓN DEL REPORTE ESTÁ LIBRE DE ERRORES DE ORTOGRAFÍA.	Sí
SE ENTREGÓ EN TIEMPO Y FORMA LA PRÁCTICA.	Sí
INCLUYE LA DOCUMENTACIÓN GENERADA CON JAVADOC.	Sí
INCLUYE EL CÓDIGO AGREGADO EN FORMATO UML.	Sí
INCLUYE LAS CAPTURAS DE PANTALLA DEL PROGRAMA FUNCIONANDO.	Sí
LA PRÁCTICA ESTÁ TOTALMENTE REALIZADA (ESPECIFIQUE EL PORCENTAJE COMPLETADO).	100%
Observaciones:	

Conclusión:

Esta práctica es para hacer la clase ArregloOrdenado y agregarle funcionalidad heredando de ArregloDatos y sobrescribiendo varios los métodos necesarios para poder indicar el orden y el tipo de orden que tendrá el arreglo. Los arreglos ordenados funcionan mucho para buscar elementos de una forma mas eficiente ya que si existe o no existe el elemento a buscar, el método termina cuando encuentra la posición que esta o debería estar el elemento y no sigue buscando.