INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	*Fundamentos de la Programación 2*				
TÍTULO DE LA PRÁCTICA:	*Arreglo de Objetos*				
NÚMERO DE PRÁCTICA:	*3*	**AÑO LECTIVO:**	*2024*	**NRO. SEMESTRE:**	*2*
FECHA DE PRESENTACIÓN	*05/10/2024*	**HORA DE PRESENTACIÓN**	*11/59/00*		
INTEGRANTE (s) *Auccacusi Conde Brayan Carlos*				**NOTA (0-20)**	
DOCENTE(s): *Ing. Lino Pinto Oppe*					

RESULTADOS Y PRUEBAS

I. EJERCICIOS RESUELTOS:

Ejercicio 1

Analice, complete y pruebe el Código de la clase DemoBatalla

```java
package L3;

public class Nave {
    private String nombre;
    private int fila;
    private String columna;
    private boolean estado;
    private int puntos;
    // Metodos mutadores
    public void setNombre( String n){
    nombre = n;
    }
    public void setFila(int f){
    fila = f;
    }
    public void setColumna(String c){
    columna = c;
    }
    public void setEstado(boolean e){
    estado = e;
    }
    public void setPuntos(int p){
    puntos = p;
    }
    // Metodos accesores
    public String getNombre(){
    return nombre;
    }
    public int getFila(){
    return fila;
    }
    public String getColumna(){
    return columna;
    }
    public boolean getEstado(){
    return estado;
    }
    public int getPuntos(){
    return puntos;
    }
    // Completar con otros métodos necesarios
}
```

```java
package L3;
import java.util.*;
public class demoBatalla {
    public static void main(String [] args){
    Nave [] flota = new Nave[10];
    Scanner sc = new Scanner(System.in);
    String nomb, col;
    int fil, punt;
    boolean est;
    for (int i = 0; i < flota.length; i++) {
        System.out.println("\nNave " + (i + 1));
        System.out.print("Nombre: ");
        nomb = sc.next();
        System.out.print("Fila: ");
        fil = sc.nextInt();
        System.out.print("Columna: ");
        col = sc.next();
        System.out.print("Estado: ");
        est = sc.nextBoolean();
        System.out.print("Puntos: ");
        punt = sc.nextInt();
        flota[i] = new Nave(); //Se crea un objeto Nave y se asigna su referencia a misNaves
        flota[i].setNombre(nomb);
        flota[i].setFila(fil);
        flota[i].setColumna(col);
        flota[i].setEstado(est);
        flota[i].setPuntos(punt);
    }
    System.out.println("\nNaves creadas:");
    mostrarNaves(flota);
    System.out.println();
    System.out.print("Ingrese el nombre de la nave(s) a mostrar: ");
    String elNombre = sc.next();
    sc.nextLine();
    mostrarPorNombre(flota, elNombre);
    System.out.print("\nIngrese el limite de puntos de la(s) nave(s) a mostrar: ");
    int puntosPedidos = sc.nextInt();
    sc.nextLine();

    mostrarPorPuntos(flota, puntosPedidos);
    System.out.println("\nLa nave con mayor número de puntos es: ");
    mostrarDatosNave(mostrarMayorPuntos(flota));
    System.out.println();
    Nave[] arrAleatorio = devuelveArray(flota);
    System.out.println("Aleatorio");
    mostrarNaves(arrAleatorio);

    }
    //Método para mostrar todas las naves
    public static void mostrarNaves(Nave[] flota){
        for (Nave misNaves : flota){
            System.out.println("\tNombre: "+ misNaves.getNombre() + "\tFila: "
            + misNaves.getFila() + "\tColumna: " + misNaves.getColumna() +
            "\tEstado: " + misNaves.getEstado() + "\t Puntos: " + misNaves.getPuntos());
        }

    }
    //Método para mostrar todas las naves de un nombre que se pide por teclado
    public static void mostrarPorNombre(Nave[] flota, String nombre){
        for (int i = 0; i < flota.length; i++) {
            if(flota[i].getNombre().equalsIgnoreCase(nombre)){
                System.out.println("\tNombre: "+ flota[i].getNombre() + "\tFila: "
                + flota[i].getFila() + "\tColumna: " + flota[i].getColumna() +
                "\tEstado: " + flota[i].getEstado() + "\t Puntos: " + flota[i].getPuntos());

            }
        }
    }
```

```java
    //Método para mostrar todas las naves con un número de puntos inferior o igual
     //al número de puntos que se pide por teclado
     public static void mostrarPorPuntos(Nave [] flota, int puntosPedidos){
         for (int i = 0; i < flota.length; i++) {
             if(flota[i].getPuntos() <= puntosPedidos)
                 System.out.println("\tNombre: "+ flota[i].getNombre() + "\tFila: "
                 + flota[i].getFila() + "\tColumna: " + flota[i].getColumna() +
                 "\tEstado: " + flota[i].getEstado() + "\t Puntos: " + flota[i].getPuntos());
         }
     }
     //Método que devuelve la Nave con mayor número de Puntos
     public static Nave mostrarMayorPuntos(Nave [] flota){
         int indexMayor = 0;
         for (int i = 0; i < flota.length; i++) {
             if(flota[i].getPuntos() > flota[indexMayor].getPuntos())
                 indexMayor = i;
         }
         return flota[indexMayor];
     }
     //Crear un método que devuelva un nuevo arreglo de objetos con todos los
     //objetos previamente ingresados
     //pero aleatoriamente desordenados
     public static Nave[] devuelveArray(Nave[] flota) {
         Random rand = new Random();
         int r1, r2;
         Nave auxiliar;
         Nave[] arrAleatorio = new Nave[flota.length];
         for (int i = 0; i < flota.length; i++) {
             arrAleatorio[i] = flota[i];
         }
         for(int i = 0; i < arrAleatorio.length; i++) {
             r1 = rand.nextInt(arrAleatorio.length);
             r2 = rand.nextInt(arrAleatorio.length);
             auxiliar = arrAleatorio[r1];
             arrAleatorio[r1] = arrAleatorio[r2];
             arrAleatorio[r2] = auxiliar;
         }
         return arrAleatorio;
     }
     public static void mostrarDatosNave(Nave naveParticular) {
         System.out.println("\tNombre: "+ naveParticular.getNombre() + "\tFila: "
          + naveParticular.getFila() + "\tColumna: " + naveParticular.getColumna() +
           "\tEstado: " + naveParticular.getEstado() + "\t Puntos: "
           + naveParticular.getPuntos());
     }
}
```

Ejercicio 2

Escribir un programa donde se creen 5 soldados considerando su nombre y nivel de vida. Ingresar sus datos y después mostrarlos.

Restricción: aplicar arreglos estándar. (Usando objetos))

```java
package L3;

public class DatosSoldados {
    public String nombre;
    public int vida;

    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public int getVida() {
        return vida;
    }
    public void setVida(int vida) {
        this.vida = vida;
    }

}

```

```java
package L3;

import java.util.*;
public class exercise2 {
    public static void main(String[] args) {
    DatosSoldados[] soldados = new DatosSoldados[5];
        for (int  i = 0; i < soldados.length; i++) {
            soldados[i] = new DatosSoldados();
        }
        recibirDatos(soldados);
        imprimir(soldados);

    }
```

```
14      public static void recibirDatos(DatosSoldados[] soldados) {
15          Scanner sc = new Scanner(System.in);
16          for (int i = 0; i < soldados.length; i++) {
17              System.out.println("Ingrese el nombre y vida del soldado " + (i + 1) + ": ");
18              System.out.print("\tNombre: ");
19              soldados[i].setNombre(sc.nextLine());
20              System.out.print("\tVida (1 - 5): ");
21              soldados[i].setVida(sc.nextInt());
22              sc.nextLine();
23
24          }
25      }
26      public static void imprimir(DatosSoldados[] soldados) {
27          System.out.println("\nSoldado\t|\tVida");
28          System.out.println("---------------------");
29          for (int i = 0; i < soldados.length; i++) {
30              System.out.println(soldados[i].getNombre() + "    |\t" + soldados[i].getVida());
31              System.out.println("---------------------");
32          }
33      }
34  }
35
36
37
38
```

Ejercicio 3

Escribir un programa donde se creen 2 ejércitos, cada uno con un número aleatorio de soldados entre 1 y 5, considerando sólo su nombre. Sus datos se inicializan automáticamente con nombres tales como “Soldado0”,“Soldado1”, etc.

Luego de crear los 2 ejércitos se deben mostrar los datos de todos los soldados de ambos ejércitos e indicar qué ejército fue el ganador.

Restricción: aplicar arreglos estándar y métodos para inicializar los ejércitos, mostrar ejército y mostrar ejército ganador. La métrica a aplicar para indicar el ganador es el mayor número de soldados de cada ejército, puede haber empates.

```java
package L3;
import java.util.Random;
public class exercise3 {
    public static void main(String[] args) {
        Random rand = new Random();
        DatosSoldados[] ejercito1 = new DatosSoldados[rand.nextInt(5) + 1];
        DatosSoldados[] ejercito2 = new DatosSoldados[rand.nextInt(5) + 1];
        crearObjetos(ejercito1);
        crearObjetos(ejercito2);
        darNombres(ejercito1,'A');
        darNombres(ejercito2, 'B');
        System.out.println("Soldados del ejercito A:");
        imprimir(ejercito1);
        System.out.println("\nSoldados del ejercito B:");
        imprimir(ejercito2);
        ganador(ejercito1, ejercito2);


    }
    public static void darNombres(DatosSoldados[] ejercitoN, char distintivo) {
        for(int i = 0; i < ejercitoN.length; i++) {
            ejercitoN[i].setNombre("Soldado " + (i + 1) + distintivo);
        }
    }
    public static void crearObjetos(DatosSoldados[] ejercitoN) {
        for(int i = 0; i < ejercitoN.length; i++) {
            ejercitoN[i] = new DatosSoldados();
        }
    }
    public static void imprimir(DatosSoldados[] soldados) {
        for(int i = 0; i < soldados.length; i++) {
            System.out.println("    " + soldados[i].getNombre());
        }
    }
    public static void ganador(DatosSoldados[] ejercito1, DatosSoldados[] ejercito2) {
        if (ejercito1.length > ejercito2.length)
            System.out.println("\nEl ejercito A gana");
        else if (ejercito1.length < ejercito2.length)
            System.out.println("\nEl ejercito B gana");
        else
            System.out.println("\nEmpate");
    }
}
```

NOTA: En el ejercicio 3 se usó la clase creada para el ejercicio 2

II. PRUEBAS

¿Con que valores comprobaste que tu práctica estuviera correcta? ¿Qué resultado esperabas obtener para cada valor de entrada? ¿Qué valor o comportamiento obtuviste para cada valor de entrada?

Ejercicio 1

```
Nave 1
Nombre: Brayan
Fila: 1
Columna: 1
Estado: true
Puntos: 1

Nave 2
Nombre: Carlos
Fila: 2
Columna: 2
Estado: false
Puntos: 2

Nave 3
Nombre: Daniel
Fila: 3
Columna: 3
Estado: true
Puntos: 3

Nave 4
Nombre: Mateo
Fila: 2
Columna: 1
Estado: false
Puntos: 5

Nave 5
Nombre: Sarai
Fila: 7
Columna: 8
Estado: true
Puntos: 10
```

```
Nave 6
Nombre: Evelyn
Fila: 4
Columna: 5
Estado: false
Puntos: 6

Nave 7
Nombre: David
Fila: 4
Columna: 7
Estado: true
Puntos: 7

Nave 8
Nombre: Marcos
Fila: 9
Columna: 1
Estado: false
Puntos: 2

Nave 9
Nombre: Madara
Fila: 2
Columna: 9
Estado: true
Puntos: 2

Nave 10
Nombre: brigith
Fila: 10
Columna: 10
Estado: true
Puntos: 10
```

```
Naves creadas:
        Nombre: Brayan  Fila: 1 Columna: 1      Estado: true     Puntos: 1
        Nombre: Carlos  Fila: 2 Columna: 2      Estado: false    Puntos: 2
        Nombre: Daniel  Fila: 3 Columna: 3      Estado: true     Puntos: 3
        Nombre: Mateo   Fila: 2 Columna: 1      Estado: false    Puntos: 5
        Nombre: Sarai   Fila: 7 Columna: 8      Estado: true     Puntos: 10
        Nombre: Evelyn  Fila: 4 Columna: 5      Estado: false    Puntos: 6
        Nombre: David   Fila: 4 Columna: 7      Estado: true     Puntos: 7
        Nombre: Marcos  Fila: 9 Columna: 1      Estado: false    Puntos: 2
        Nombre: Madara  Fila: 2 Columna: 9      Estado: true     Puntos: 2
        Nombre: brigith Fila: 10        Columna: 10     Estado: true     Puntos: 10
```

```
Ingrese el nombre de la nave(s) a mostrar: brayan
        Nombre: Brayan  Fila: 1 Columna: 1      Estado: true     Puntos: 1

Ingrese el limite de puntos de la(s) nave(s) a mostrar: 5
        Nombre: Brayan  Fila: 1 Columna: 1      Estado: true     Puntos: 1
        Nombre: Carlos  Fila: 2 Columna: 2      Estado: false    Puntos: 2
        Nombre: Daniel  Fila: 3 Columna: 3      Estado: true     Puntos: 3
        Nombre: Mateo   Fila: 2 Columna: 1      Estado: false    Puntos: 5
        Nombre: Marcos  Fila: 9 Columna: 1      Estado: false    Puntos: 2
        Nombre: Madara  Fila: 2 Columna: 9      Estado: true     Puntos: 2

La nave con mayor número de puntos es:
        Nombre: Sarai   Fila: 7 Columna: 8      Estado: true     Puntos: 10

Aleatorio
        Nombre: Marcos  Fila: 9 Columna: 1      Estado: false    Puntos: 2
        Nombre: Carlos  Fila: 2 Columna: 2      Estado: false    Puntos: 2
        Nombre: Brayan  Fila: 1 Columna: 1      Estado: true     Puntos: 1
        Nombre: Mateo   Fila: 2 Columna: 1      Estado: false    Puntos: 5
        Nombre: Madara  Fila: 2 Columna: 9      Estado: true     Puntos: 2
        Nombre: brigith Fila: 10        Columna: 10     Estado: true     Puntos: 10
        Nombre: Daniel  Fila: 3 Columna: 3      Estado: true     Puntos: 3
        Nombre: David   Fila: 4 Columna: 7      Estado: true     Puntos: 7
        Nombre: Evelyn  Fila: 4 Columna: 5      Estado: false    Puntos: 6
        Nombre: Sarai   Fila: 7 Columna: 8      Estado: true     Puntos: 10
PS C:\Users\Hogar\Documents\BRAYAN\EP2 - Laboratories>
```

- *Ingrese los datos de cada nave como su nombre, fila, columna, estado y puntos.*
- *Puse una parte para que devuelva los datos de la nave por nombre, probe con un nombre antes ingresado, espere que me devuelva la nave que coincidiera.*
- *Ingrese un limite de puntos, luego esperaba que me de todas las naves que tengas menos o igual que esos puntos.*
- *Espere que me de la nave con mas puntos.*

Ejercicio 2

```
Ingrese el nombre y vida del soldado 1:
        Nombre: Brayan
        Vida (1 - 5): 2
Ingrese el nombre y vida del soldado 2:
        Nombre: Carlos
        Vida (1 - 5): 4
Ingrese el nombre y vida del soldado 3:
        Nombre: Daniel
        Vida (1 - 5): 1
Ingrese el nombre y vida del soldado 4:
        Nombre: Marcos
        Vida (1 - 5): 4
Ingrese el nombre y vida del soldado 5:
        Nombre: Mateo5
        Vida (1 - 5): 5

Soldado |       Vida
--------------------
Brayan  |       2
--------------------
Carlos  |       4
--------------------
Daniel  |       1
--------------------
Marcos  |       4
--------------------
Mateo5  |       5
--------------------
```

- *Espere que me de la interfaz para ingresar el nombre y la vida de cada soldado*
- *Ingrese el nombre y la vida. Espere a que me de los datos (nombre y vida) en formato tabla.*

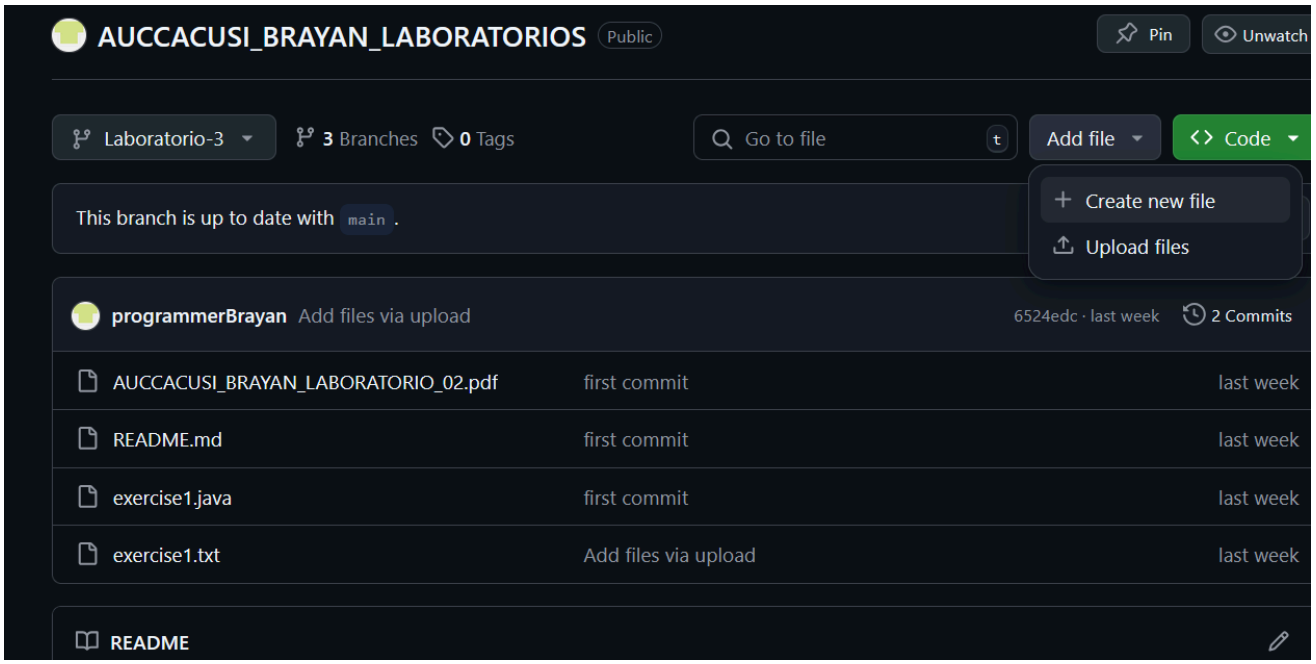
Ejercicio 3

```
Soldados del ejercito A:
    Soldado 1A
    Soldado 2A
    Soldado 3A
    Soldado 4A

Soldados del ejercito B:
    Soldado 1B
    Soldado 2B
    Soldado 3B
    Soldado 4B
    Soldado 5B

El ejercito B gana
```

- *Como esperaba no tuve que ingresar nada porque la cantidad de soldados y sus nombres se generan automáticamente en el programa.*
- *Esperaba que se imprima el nombre del ejército y de cada soldado, para cada ejército habría una variación para distinguirlo, además entre uno y otro el nombre solo variaría en un carácter para diferenciarlos (nombrarlos).*



Enlace a mi repositorio:

https://github.com/programmerBrayan/AUCCACUSI_BRAYAN_LABORATORIO_02.git

III. CUESTIONARIO:

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	x	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	x	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	x	2	

4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	x	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	x	2	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	x	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	x	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	x	3	
TOTAL		20	7	18	

CONCLUSIONES

El array de objetos es una herramienta poderosa ya que podemos almacenar diferentes tipos de datos,como String, int, etc. Esto no ahorra el crear diferentes arrays para cada tipo de dato además de que nos permite trabajar ordenadamente.
Para trabajar con un arreglo de objetos, debemos crear la clase de esos objetos, además hay que considerar que los arreglos de objetos tienen una sintaxis más avanzada (debemos especificar un poco más a la hora de trabajar)
Ejemplo en un array normal basta con colocar el nombre del array + el índice para obtener un dato, en un array de objetos debemos colocar a demás de lo anterior los métodos creados en la clase, como los getter o setters..

METODOLOGÍA DE TRABAJO

1. *Primero leí y traté de comprender lo que me pedían además del código a corregir.*
2. *Investigue de proyectos similares en la internet además de que se trataba el problema a solucionar.*

3. *Escribí el código y compile..*
4. *Probe y corregi los errores que hubiera*
5. *Le di un buen estilo para cuando imprimiera.*

REFERENCIAS Y BIBLIOGRAFÍA

https://www.youtube.com/watch?v=Q1cP-yugi5M

https://github.com/programmerBrayan/AUCCACUSI_BRAYAN_LABORATORIO_02.git