Universidad Nacional de San Agustín
Escuela Profesional de Ingeniería de Sistemas
Fundamentos de Programación II
Practica de Laboratorio 3:
Arreglos de Objetos

I

OBJETIVOS

- Crear e inicializar arreglos de objetos
- Solucionar problemas

II

MARCO TEORICO

Arreglos de Objetos

Un arreglo representa una colección de elementos del mismo tipo bajo un mismo nombre. Un arreglo no sólo almacena tipos primitivos (enteros, reales, booleanos, etc), sino también puede almacenar objetos. En la Figura 1 se muestra la clase `Persona`, con sus atributos y sus métodos.

Persona
String nombre int edad char genero
void setNombre(String elNombre) void setEdad(int laEdad) void setGenero(char elGenero) String getNombre() int getEdad() char getGenero()

Figura 1: Clase Persona

En la Figura 2 se muestra el código para generar un arreglo que me permita almacenar 20 objetos del tipo Persona. Luego de crear el arreglo se está asignado el primer objeto a la primera posición del arreglo.

```
Persona[ ] misAmigos;

misAmigos = new Persona[20];

misAmigos[0] = new Persona( );

misAmigos[0].setNombre("Juancito");
```

Figura 2: Ejemplo de creación de un arreglo de objetos

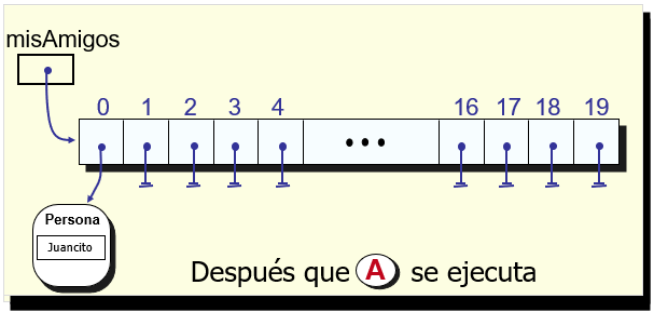
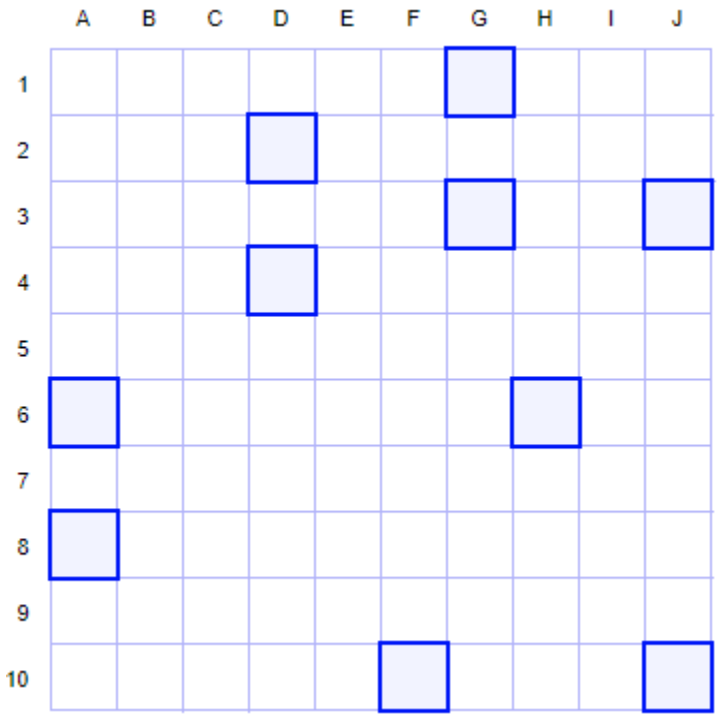


Figura 3: Representación gráfica del arreglo de objetos.

III ACTIVIDADES

Cree un Proyecto llamado Laboratorio3
Usted deberá agregar las clases Nave.java y DemoBatalla.java.

1. Analice, complete y pruebe el Código de la clase DemoBatalla



```java
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
}
```

```
import java.util.*;
public class DemoBatalla {

	public static void main(String [] args){
		Nave [] misNaves = new Nave[10];
		Scanner sc = new Scanner(System.in);
		String nomb, col;
		int fil, punt;
		boolean est;

		for (int i = 0; i < misNaves.length; i++) {
			System.out.println("Nave " + (i+1));
			System.out.print("Nombre: ");
			nomb = sc.next();
			System.out.println("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();

			misNaves[i] = new Nave(); //Se crea un objeto Nave y se asigna su referencia a misNaves

			misNaves[i].setNombre(nomb);
			misNaves[i].setFila(fil);
			misNaves[i].setColumna(col);
			misNaves[i].setEstado(est);
			misNaves[i].setPuntos(punt);
		 }

		System.out.println("\nNaves creadas:");
		mostrarNaves(misNaves);
		mostrarPorNombre(misNaves);
		mostrarPorPuntos(misNaves);
		System.out.println("\nNave con mayor número de puntos: " + mostrarMayorPuntos(misNaves));
	}
	//Método para mostrar todas las naves
	public static void mostrarNaves(Nave [] flota){
	}

	//Método para mostrar todas  las naves de un nombre que se pide por teclado
	public static void mostrarPorNombre(Nave [] flota){
	}

	//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){
	}

	//Método que devuelve la Nave con mayor número de Puntos
	public static Nave mostrarMayorPuntos(Nave [] flota){
	}

	//Crear un método que devuelva un nuevo arreglo de objetos con todos los objetos previamente ingresados
	//pero aleatoriamente desordenados

}
```

2. Solucionar la Actividad 4 de la Práctica 1 pero usando arreglo de objetos
3. Solucionar la Actividad 5 de la Práctica 1 pero usando arreglos de objetos