

	<p align="center"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
<b>Aprobación:</b> 2022/03/01	<b>Código:</b> GUIA-PRLE-001	<b>Página:</b> 1

## INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
<b>ASIGNATURA:</b>	<i>Fundamentos de la Programación 2</i>				
<b>TÍTULO DE LA PRÁCTICA:</b>	<i>Practica de Laboratorio 5: Arreglos bidimensionales de Objetos</i>				
<b>NÚMERO DE PRÁCTICA:</b>	<i>5</i>	<b>AÑO LECTIVO:</b>	<i>2024</i>	<b>NRO. SEMESTRE:</b>	<i>Segundo</i>
<b>FECHA DE PRESENTACIÓN</b>	<i>18/10/2024</i>	<b>HORA DE PRESENTACIÓN</b>	<i>18:30</i>		
<b>INTEGRANTE (s)</b> <i>Santiago Alonso Quintanilla Chávez</i>				<b>NOTA (0-20)</b>	
<b>DOCENTE(s):</b> <i>Ing. Lino Jose Pinto Oppe</i>					

RESULTADOS Y PRUEBAS
<p><b>I. EJERCICIOS RESUELTOS:</b></p> <ol style="list-style-type: none"> <li>1. Cree un Proyecto llamado Laboratorio5</li> <li>2. Usted deberá crear las dos clases Soldado.java y VideoJuego2.java. Puede reutilizar lo desarrollado en Laboratorio 3 y 4.</li> <li>3. Del Soldado nos importa el nombre, nivel de vida, fila y columna (posición en el tablero).</li> <li>4. El juego se desarrollará en el mismo tablero de los laboratorios anteriores. Pero ahora el tablero debe ser un arreglo bidimensional de objetos.</li> <li>5. Inicializar el tablero con n soldados aleatorios entre 1 y 10. Cada soldado tendrá un nombre autogenerado: Soldado0, Soldado1, etc., un valor de nivel de vida autogenerado aleatoriamente [1..5], la fila y columna también autogenerados aleatoriamente (verificar que no puede haber 2 soldados en el mismo cuadrado). Se debe mostrar el tablero con todos los soldados creado (usar caracteres como   _ y otros). Además, mostrar los datos del Soldado con mayor nivel de vida, el promedio de nivel de vida de todos los soldados creados, el nivel de vida de todo el ejército, los datos de todos los soldados en el orden que fueron creados y un ranking de poder de todos los soldados creados, del que tiene más nivel de vida al que tiene menos (usar al menos 2 algoritmos de ordenamiento).             <ol style="list-style-type: none"> <li>a. Código de la clase "Soldado":</li> </ol> </li> </ol>

```
1 public class Soldado {
2     private String nombre;
3     private int nivelVida;
4     private int fila;
5     private int columna;
6     private boolean ocupado;
7
8     public void setNombre(String n){
9         nombre=n;
10    }
11    public void setNivelVida(int v) {
12        nivelVida=v;
13    }
14    public void setFila(int f) {
15        fila=f;
16    }
17    public void setColumna (int c) {
18        columna=c;
19    }
20    public void setOcupado (boolean o) {
21        ocupado=o;
22    }
23    public String getNombre(){
24        return nombre;
25    }
26    public int getNivelVida() {
27        return nivelVida;
28    }
29    public int getFila() {
30        return fila;
31    }
32    public int getColumna() {
33        return columna;
34    }
35    public boolean getOcupado() {
36        return ocupado;
37    }
38    public String toString() {
39        return "-Nombre: "+nombre+"\t-Posición: ("+fila+","+columna+")\t-Nivel de Vida: "+nivelVida;
40    }
41 }
```

**b. Código de clase “Videojuego2”:**

```
1 import java.util.*;
2 public class Videojuego2 {
3     public static void main(String[] args) {
4         Random rand=new Random();
5         int NumSold=rand.nextInt(10)+1;
6         Soldado [][] tablero=Inicializar(NumSold);
7         imprimirTablero(tablero);
8         Soldado [] ejercito=arregloEjercito(tablero, NumSold);
9         Soldado soldMayor=soldadoMayorVida(ejercito);
10        System.out.println("El soldado con mayor vida es: \n"+soldMayor);
11        Double vida=promedioNivelVida(ejercito);
12        System.out.println("El promedio del nivel de vida es: "+vida);
13        imprimirVida(ejercito);
14
15        imprimirOrdenCreacion(ejercito);
16        imprimirRankingVida(ejercito);
17    }
18    //Metodo que imprime el tablero, donde las 'x' marcan los soldados creados//
19    public static void imprimirTablero(Soldado [][]arreglo) {
20        System.out.println("\n_____");
21        System.out.println("Tablero de posiciones: ");
22        for (int k=0;k<arreglo.length;k++) {
23            System.out.println("");
24            for (int j=0;j<arreglo[k].length;j++) {
25                boolean ocupado=(arreglo[k][j]).getOcupado();
26                if (ocupado) {
27                    System.out.print("|_X_|");
28                } else {
29                    System.out.print("|___|");
30                }
31            }
32        }
33    }
34    //Metodo que inicializa el arreglo de soldados y crea la cantidad de soldados segun el
35    //numero aleatorio generado en el metodo main//
36    public static Soldado[][] inicializar(int x) {
37        Random rand=new Random();
38        Soldado ejercito[][]=new Soldado[10][10];
39        for (int i=0;i<ejercito.length;i++) {
40            for (int j=0;j<ejercito[i].length;j++) {
41                ejercito[i][j]=new Soldado();
42                ejercito[i][j].setOcupado(false);
43            }
44        }
45    }
46 }
```

```

45     for (int k=0;k<x;k++) {
46         int fila=rand.nextInt(10);
47         int columna=rand.nextInt(10);
48         boolean ocupado=(ejercito[fila][columna]).getOcupado();
49         while (ocupado) {
50             fila=rand.nextInt(10);
51             columna=rand.nextInt(10);
52             ocupado=ejercito[fila][columna].getOcupado();
53         }
54         String nombre="Soldado"+(k+1);
55         ejercito[fila][columna].setNombre(nombre);
56         ejercito[fila][columna].setFila(fila+1);
57         ejercito[fila][columna].setColumna(columna+1);
58         ejercito[fila][columna].setNivelVida(rand.nextInt(5)+1);
59         ejercito[fila][columna].setOcupado(true);
60     }
61     return ejercito;
62 }
63 //Metodo que busca el soldado con mayor nivel de vida y retorna el objeto//
64 public static Soldado soldadoMayorVida (Soldado [] arreglo) {
65     int mayor=0;
66     Soldado soldadoMayor=null;
67     System.out.println("\n_____");
68     for (int i=0;i<arreglo.length;i++) {
69         int vida=arreglo[i].getNivelVida();
70         if (vida>mayor) {
71             mayor=vida;
72             soldadoMayor=arreglo[i];
73         }
74     }
75     return soldadoMayor;
76 }
77 //Metodo que muestra el promedio del nivel de vida de todos los soldados creados//
78 public static double promedioNivelVida (Soldado [] arreglo) {
79     int acumulado=0;
80     System.out.println("\n_____");
81     for (int i=0;i<arreglo.length;i++) {
82         boolean ocupado=arreglo[i].getOcupado();
83         if (ocupado) {
84             int vida=arreglo[i].getNivelVida();
85             acumulado+=vida;
86         }
87     }

```

```

88         double promedio=acumulado/(arreglo.length);
89         return promedio;
90     }
91     //Metodo que muestra el nivel de vida de todo el ejercito//
92     public static void imprimirVida(Soldado[] arreglo) {
93         System.out.println("\n_____");
94         System.out.println("Niveles de vida de todo el ejercito: ");
95         for (int i=0;i<arreglo.length;i++) {
96             System.out.println("-Soldado: "+arreglo[i].getNombre()+"\tvvida: "
97             +arreglo[i].getNivelVida());
98         }
99     }
100     public static Soldado[] arregloEjercito (Soldado[][] arreglo, int x) {
101         Soldado ejercito[]=new Soldado[x];
102         int contador=0;
103         for (int i=0;i<arreglo.length;i++) {
104             for (int j=0;j<arreglo[i].length;j++) {
105                 boolean ocupado=arreglo[i][j].getOcupado();
106                 if (ocupado) {
107                     ejercito[contador]=arreglo[i][j];
108                     contador++;
109                 }
110             }
111         }
112         return ejercito;
113     }
114     //Metodo que ordena a los soldados en el orden que fueron creados
115     //y los imprime en tal orden
116     //--->Se emplea el algoritmo de Ordenamiento Burbuja
117     public static void imprimirOrdenCreacion (Soldado[] arreglo){
118         System.out.println("\n_____");
119         System.out.println("Orden de los Soldados segun su creacion: ");
120         for (int i=1;i<arreglo.length;i++) {
121             for (int j=0;j<arreglo.length-1;j++) {
122                 if ((arreglo[j].getNombre()).compareTo(arreglo[j+1].getNombre())>0) {
123                     Soldado temp=arreglo[j];
124                     arreglo[j]=arreglo[j+1];
125                     arreglo[j+1]=temp;
126                 }
127             }
128         }
129         for (int k=0;k<arreglo.length;k++) {
130             System.out.println(arreglo[k]);
131         }
132     }

```

```
-Soldado: Soldado4      Vida: 5
-Soldado: Soldado5      Vida: 2
-Soldado: Soldado2      Vida: 3
-Soldado: Soldado1      Vida: 2
-Soldado: Soldado3      Vida: 5
-Soldado: Soldado6      Vida: 5
```

```

Orden de los Soldados segun su creacion:
-Nombre: Soldado1      -Posición: (7,5)      -Nivel de Vida: 2
-Nombre: Soldado2      -Posición: (7,3)      -Nivel de Vida: 3
-Nombre: Soldado3      -Posición: (8,7)      -Nivel de Vida: 5
-Nombre: Soldado4      -Posición: (4,9)      -Nivel de Vida: 5
-Nombre: Soldado5      -Posición: (6,1)      -Nivel de Vida: 2
-Nombre: Soldado6      -Posición: (8,10)     -Nivel de Vida: 5

Ranking del nivel de vida de los Soldados:
-Nombre: Soldado3      -Posición: (8,7)      -Nivel de Vida: 5
-Nombre: Soldado4      -Posición: (4,9)      -Nivel de Vida: 5
-Nombre: Soldado6      -Posición: (8,10)     -Nivel de Vida: 5
-Nombre: Soldado2      -Posición: (7,3)      -Nivel de Vida: 3
-Nombre: Soldado5      -Posición: (6,1)      -Nivel de Vida: 2
-Nombre: Soldado1      -Posición: (7,5)      -Nivel de Vida: 2
  
```

- **Evidencia de los Commits:**

Commit changes

Commit message

Create Videojuego2

Extended description

Add an optional extended description..

☒ Commit directly to the main branch
 ☐ Create a new branch for this commit and start a pull request [Learn more about pull requests](#)

Cancel

Commit changes

10:25:25 a. m.

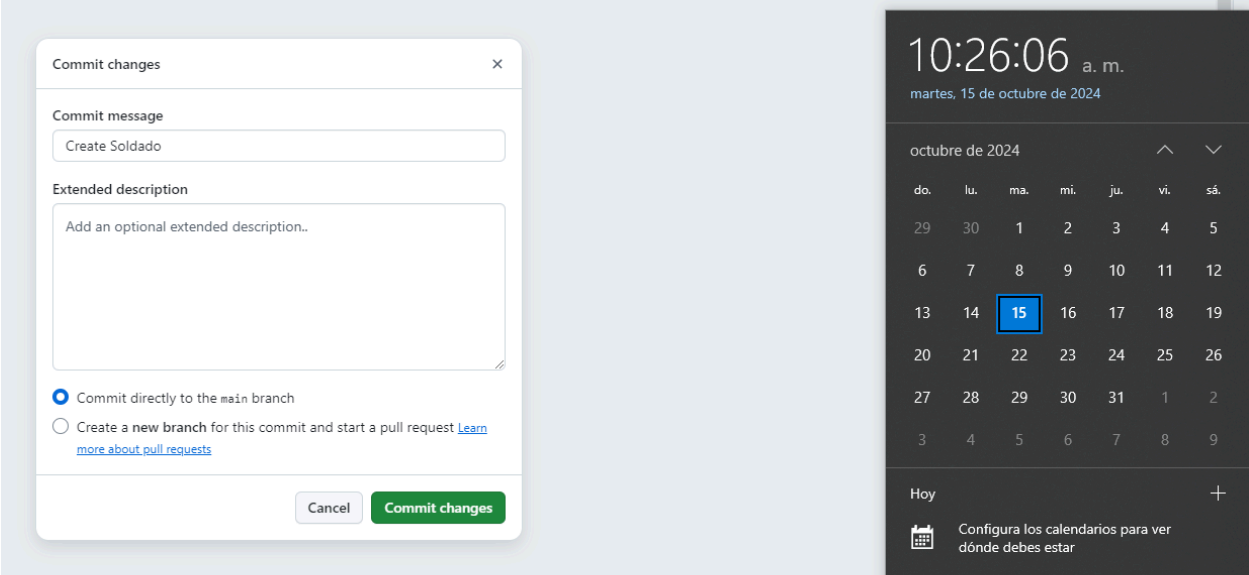
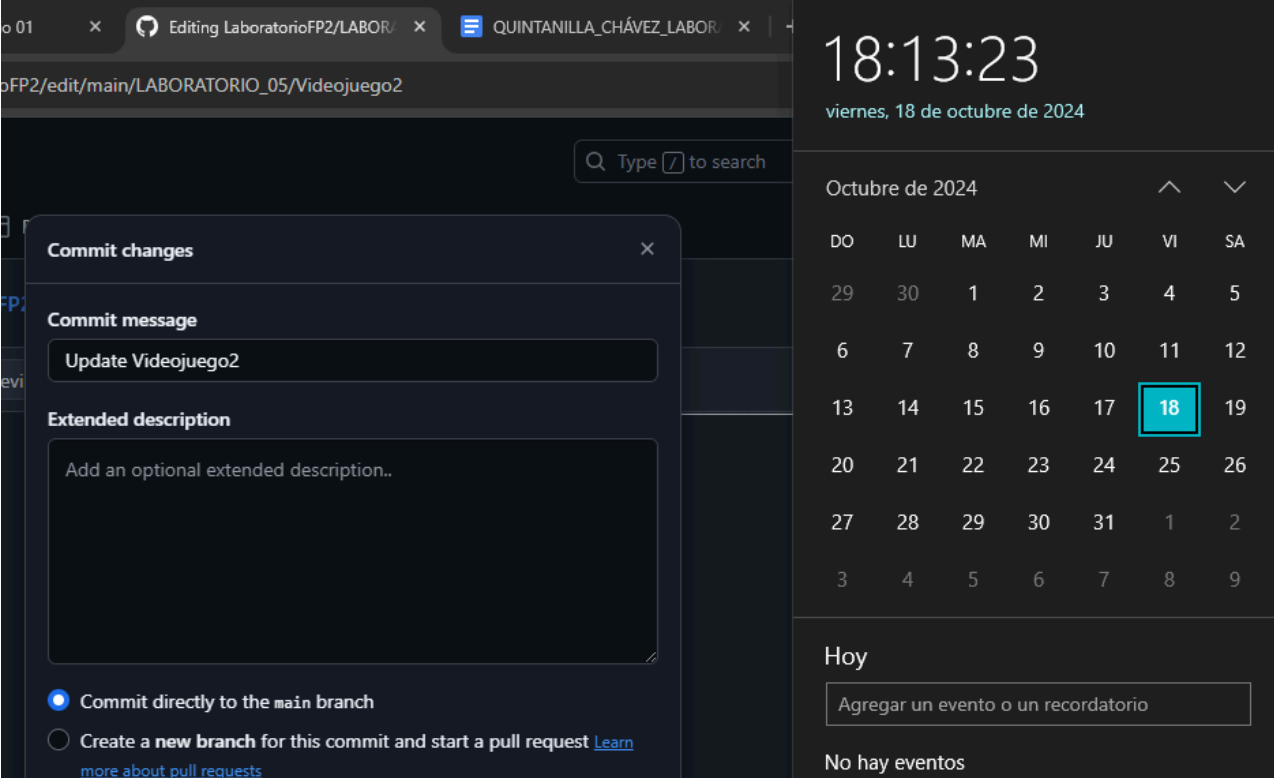
martes, 15 de octubre de 2024

octubre de 2024

do.	lu.	ma.	mi.	ju.	vi.	sá.
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

Hoy

Configura los calendarios para ver dónde debes estar

- **Link al repositorio de GitHub - Laboratorio05:**  
[https://github.com/SantiagoQuintanilla/LaboratorioFP2/tree/main/LABORATORIO\\_05](https://github.com/SantiagoQuintanilla/LaboratorioFP2/tree/main/LABORATORIO_05)



● **Autoevaluación:**

	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. <del>Commits</del>	Hay capturas de pantalla de los <del>commits</del> más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4		X	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2		X	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2		X	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		X	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2		X	
7. Ortografía	El documento no muestra errores ortográficos.	2		X	
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			
TOTAL		20			