





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

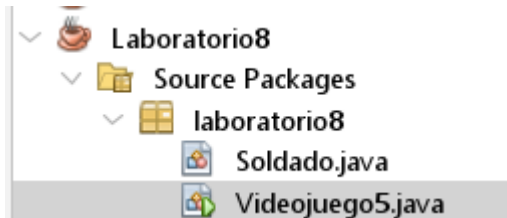
INFORME DE LABORATORIO

(formato estudiante)

| INFORMACIÓN BÁSICA | | | | | |
|--|--|-----------------------------|-------|-----------------------|----|
| ASIGNATURA: | LABORATORIO DE FUNDAMENTOS DE PROGRAMACIÓN 2 | | | | |
| TÍTULO DE LA PRÁCTICA: | HashMap | | | | |
| NÚMERO DE PRÁCTICA: | 08 | AÑO LECTIVO: | 2024 | NRO. SEMESTRE: | II |
| FECHA DE PRESENTACIÓN | 29/11/2024 | HORA DE PRESENTACIÓN | 16:15 | | |
| INTEGRANTE (s) DIEGO ARISTIDES CERVANTES APAZA | | | | NOTA (0-20) | |
| DOCENTE(s): JOSE LINO PINTO OPPE | | | | | |

| RESULTADOS Y PRUEBAS |
|--|
| <p>I. EJERCICIOS RESUELTOS:</p> <ol style="list-style-type: none"> 1. Cree un Proyecto llamado Laboratorio8 2. Usted deberá crear las dos clases Soldado.java y VideoJuego5.java. Puede reutilizar lo desarrollado en Laboratorios anteriores. 3. Del Soldado nos importa el nombre, puntos de vida, fila y columna (posición en el tablero). 4. El juego se desarrollará en el mismo tablero de los laboratorios anteriores. Para crear el tablero utilice la estructura de datos más adecuada. 5. Tendrá 2 Ejércitos (usar HashMaps). Inicializar el tablero con n soldados aleatorios entre 1 y 10 para cada Ejército. Cada soldado tendrá un nombre autogenerado: Soldado0X1, Soldado1X1, etc., un valor de puntos de vida autogenerado aleatoriamente [1..5], la fila y columna también autogenerados aleatoriamente (no puede haber 2 soldados en el mismo cuadrado). Se debe mostrar el tablero con todos los soldados creados (distinguir los de un ejército de los del otro ejército). Además de los datos del Soldado con mayor vida de cada ejército, el promedio de puntos de vida de todos los soldados creados por ejército, los datos de todos los soldados por ejército en el orden que fueron creados y un ranking de poder de todos los soldados creados por ejército (del que tiene más nivel de vida al que tiene menos) usando 2 diferentes algoritmos de ordenamiento (indicar conclusiones respecto a este ordenamiento de HashMaps). Finalmente, que muestre qué ejército ganará la batalla (indicar la métrica usada para decidir al ganador de la batalla). Hacerlo como programa iterativo. |

| | | |
|--|--|---|
|  | <p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| <p align="center">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p> | | |
| <p>Aprobación: 2022/03/01</p> | <p align="center">Código: GUIA-PRLE-001</p> | <p align="right">Página: 2</p> |



CÓDIGO:

ESQUELETO DEL CÓDIGO

```
public static void main(String[] args) {
    Random rand = new Random();
    Scanner scan = new Scanner(System.in);

    boolean repetitivo = true; // Controlar el bucle

    while (repetitivo) {
        // Crear dos ejércitos (HashMaps)
        Map<Integer, Soldado> ejercito1 = new HashMap<>();
        Map<Integer, Soldado> ejercito2 = new HashMap<>();



        // Generación de filas y columnas aleatorias para el tablero
        int filas = rand.nextInt(8) + 3; // Tamaño aleatorio entre 3 y 10
        int columnas = rand.nextInt(8) + 3;

        // Crear soldados para el primer ejército (1 a 10 soldados)
        crearEjercito(rand, ejercito1, 1, filas, columnas);
        // Crear soldados para el segundo ejército (1 a 10 soldados)
        crearEjercito(rand, ejercito2, 2, filas, columnas);

        // Mostrar el tablero con los soldados de ambos ejércitos
        System.out.println("\nTablero:");
        mostrarTablero(ejercito1, ejercito2, filas, columnas);

        // Mostrar el soldado con mayor vida de cada ejército
        System.out.println("\nSoldado con mayor vida del Ejército 1: " + soldadoConMasVida(ejercito1).getNombre());
        System.out.println("Soldado con mayor vida del Ejército 2: " + soldadoConMasVida(ejercito2).getNombre());

        // Promedio de puntos de vida por ejército
        System.out.println("\nPromedio de puntos de vida Ejército 1: " + promedioPuntosDeVida(ejercito1));
        System.out.println("Promedio de puntos de vida Ejército 2: " + promedioPuntosDeVida(ejercito2));
    }
}
```

| | | |
|---|---|---|
|  | <p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| <p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p> | | |
| <p>Aprobación: 2022/03/01</p> | <p>Código: GUIA-PRLE-001</p> | <p>Página: 3</p> |

```

// Mostrar ranking de soldados por nivel de vida (por ejército)
mostrarRanking(ejercito1, "Ejército 1", rand);
mostrarRanking(ejercito2, "Ejército 2", rand);

// Decidir qué ejército gana basado en el total de puntos de vida
int totalEjercito1 = totalPuntosDeVida(ejercito1);
int totalEjercito2 = totalPuntosDeVida(ejercito2);

System.out.println("\nTotal de puntos de vida Ejército 1: " + totalEjercito1);
System.out.println("Total de puntos de vida Ejército 2: " + totalEjercito2);



// Determinar al ganador
if (totalEjercito1 > totalEjercito2) {
    System.out.println("\n¡El Ejército 1 gana!");
} else if (totalEjercito1 < totalEjercito2) {
    System.out.println("\n¡El Ejército 2 gana!");
} else {
    System.out.println("\n¡Es un empate!");
}

// Preguntar si desea seguir creando ejércitos
System.out.print("\n¿Quieres crear nuevos ejércitos? (sí/no): ");
String respuesta = scan.nextLine();
if (!respuesta.equalsIgnoreCase("sí")) {
    repetitivo = false;
    System.out.println("Gracias por jugar.");
}
}

scan.close();
}

```

CREACIÓN DE UN EJERCITO MEDIANTE HASHMAP:

| | | |
|---|---|---|
|  | <p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| <p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p> | | |
| <p>Aprobación: 2022/03/01</p> | <p>Código: GUIA-PRLE-001</p> | <p>Página: 4</p> |

```
// Método para crear un ejército de soldados aleatorios
public static void crearEjercito(Random rand, Map<Integer, Soldado> ejercito, int numeroEjercito,
                                int filas, int columnas) {
    int numeroSoldados = rand.nextInt(10) + 1; // Número de soldados entre 1 y 10
    for (int i = 0; i < numeroSoldados; i++) {
        int fila, columna;
        // Asegurarse que no haya dos soldados en la misma posición
        do {
            fila = rand.nextInt(filas);
            columna = rand.nextInt(columnas);
        } while (yaExisteSoldadoEnPosicion(ejercito, fila, columna)); // Verificar si ya hay un soldado en esa celda



        // Crear un nombre único para el soldado
        String nombre = "Soldado" + i + "X" + numeroEjercito;
        int puntosDeVida = rand.nextInt(5) + 1;

        Soldado soldado = new Soldado();
        soldado.setNombre(nombre);
        soldado.setFila(fila);
        soldado.setColumna(columna);
        soldado.setNivelDeVida(puntosDeVida);

        // Guardar el soldado en el ejército
        ejercito.put(i, soldado);
    }
}
```

VERIFICACIÓN QUE CADA SOLDADO NO SE REPITA EN CADA CELDA QUE CAE:

```
// Verificar si ya existe un soldado en la posición (fila, columna)
public static boolean yaExisteSoldadoEnPosicion(Map<Integer, Soldado> ejercito, int fila, int columna) {
    for (Soldado soldado : ejercito.values()) {
        if (soldado.getFila() == fila && soldado.getColumna() == columna) {
            return true;
        }
    }
    return false;
}
```

| | | |
|---|---|---|
|  | <p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |
| Aprobación: 2022/03/01 | Código: GUIA-PRLE-001 | Página: 5 |

EJECUCIÓN Y MUESTRA DEL TABLERO:



```
// Mostrar el tablero con los soldados de ambos ejércitos
public static void mostrarTablero(Map<Integer, Soldado> ejercito1, Map<Integer, Soldado> ejercito2,
    int filas, int columnas) {
    // Usar un tablero representado por filas y columnas
    String[][] tablero = new String[filas][columnas];
    // Inicializamos el tablero con "_" indicando una celda vacía
    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            tablero[i][j] = "_";
        }
    }

    // Colocamos los soldados de los dos ejércitos en el tablero
    for (Soldado soldado : ejercito1.values()) {
        tablero[soldado.getFila()][soldado.getColumna()] = "S1"; // "S1" para Ejército 1
    }
    for (Soldado soldado : ejercito2.values()) {
        tablero[soldado.getFila()][soldado.getColumna()] = "S2"; // "S2" para Ejército 2
    }

    // Mostrar el tablero
    for (int i = 0; i < filas; i++) {
        for (int j = 0; j < columnas; j++) {
            System.out.print("| " + tablero[i][j] + " ");
        }
        System.out.println("\n");
    }
}
```

ESCRITURA DEL SOLDADO CON MAYOR VIDA:

```
// Obtener el soldado con más puntos de vida
public static Soldado soldadoConMasVida(Map<Integer, Soldado> ejercito) {
    Soldado soldadoMax = null;
    for (Soldado soldado : ejercito.values()) {
        if (soldadoMax == null || soldado.getNivelDeVida() > soldadoMax.getNivelDeVida()) {
            soldadoMax = soldado;
        }
    }
    return soldadoMax;
}
```

| | | |
|---|---|---|
|  | <p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| <p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p> | | |
| <p>Aprobación: 2022/03/01</p> | <p>Código: GUIA-PRLE-001</p> | <p>Página: 6</p> |

CÁLCULO DEL PROMEDIO DE VIDA DEL EJÉRCITO:

```
// Calcular el promedio de puntos de vida de un ejército
public static double promedioPuntosDeVida (Map<Integer, Soldado> ejercito) {
    int suma = 0;
    int contador = 0;
    for (Soldado soldado : ejercito.values()) {
        suma += soldado.getNivelDeVida();
        contador++;
    }
    return (double) suma / contador;
}
```

CÁLCULO DEL RANKING:

```
// Mostrar el ranking de soldados por nivel de vida (por ejército)
public static void mostrarRanking (Map<Integer, Soldado> ejercito, String nombreEjercito, Random rand) {
    List<Soldado> soldados = new ArrayList<>(ejercito.values());

    // Ordenamiento con Bubble Sort
    ordenarSoldadosPorNivelDeVidaBurbuja(soldados);

    System.out.println("\nRanking de poder de " + nombreEjercito + " con Bubble Sort:");
    for (Soldado soldado : soldados) {
        System.out.println(soldado.getNombre() + " | Puntos de vida: " + soldado.getNivelDeVida());
    }

    // Ordenamiento con Selección
    ordenarSoldadosPorNivelDeVidaSeleccion(soldados);

    System.out.println("\nRanking de poder de " + nombreEjercito + " con Selección:");
    for (Soldado soldado : soldados) {
        System.out.println(soldado.getNombre() + " | Puntos de vida: " + soldado.getNivelDeVida());
    }
}
```

TOTAL DE VIDA DE LOS EJÉRCITOS:

```
// Total de puntos de vida de todos los soldados en el ejército
public static int totalPuntosDeVida(Map<Integer, Soldado> ejercito) {
    int total = 0;
    for (Soldado soldado : ejercito.values()) {
        total += soldado.getNivelDeVida();
    }
    return total;
}
```

MÉTODOS DE ORDENAMIENTO EN LOS EJÉRCITOS:

```
// Métodos de ordenamiento (Bubble Sort y Selección)
public static void ordenarSoldadosPorNivelDeVidaBurbuja(List<Soldado> soldados) {
    boolean ordenado;
    do {
        ordenado = true;
        for (int i = 0; i < soldados.size() - 1; i++) {
            if (soldados.get(i).getNivelDeVida() < soldados.get(i + 1).getNivelDeVida()) {
                // Intercambiar
                Soldado temp = soldados.get(i);
                soldados.set(i, soldados.get(i + 1));
                soldados.set(i + 1, temp);
                ordenado = false;
            }
        }
    } while (!ordenado);
}

public static void ordenarSoldadosPorNivelDeVidaSeleccion(List<Soldado> soldados) {
    for (int i = 0; i < soldados.size() - 1; i++) {
        int maxIdx = i;
        for (int j = i + 1; j < soldados.size(); j++) {
            if (soldados.get(j).getNivelDeVida() > soldados.get(maxIdx).getNivelDeVida()) {
                maxIdx = j;
            }
        }
        // Intercambiar
        Soldado temp = soldados.get(maxIdx);
        soldados.set(maxIdx, soldados.get(i));
        soldados.set(i, temp);
    }
}
```

CLASE SOLDADO:

```
package laboratorio8;

public class Soldado {
    private String nombre;
    private int fila;
    private int columna;
    private int nivelDeVida;

    // Métodos mutadores
    public void setNombre(String n) {
        nombre = n;
    }
    public void setFila(int f) {
        fila = f;
    }
    public void setColumna(int c) {
        columna = c;
    }
    public void setNivelDeVida(int f) {
        nivelDeVida = f;
    }

    // Métodos accesorios
    public String getNombre() {
        return nombre;
    }
    public int getFila() {
        return fila;
    }
    public int getColumna() {
        return columna;
    }
    public int getNivelDeVida() {
        return nivelDeVida;
    }
}
```


II. PRUEBAS

TABLERO GENERADO AL AZAR:

```

Tablero:
| _ | S1 | S1 |
| S1 | _ | _ |
| _ | _ | _ |
| S1 | _ | _ |
| S1 | _ | S2 |
| S1 | _ | _ |
| _ | S1 | S1 |
| _ | _ | _ |
| S2 | S1 | _ |
  
```

SOLDADOS CON MAYORES VIDAS GENERADAS DE CADA EJÉRCITO:

```



Soldado con mayor vida del Ejército 1: Soldado5X1
Soldado con mayor vida del Ejército 2: Soldado1X2
  
```

PROMEDIO DE VIDA POR EJÉRCITO:

```

Promedio de puntos de vida Ejército 1: 2.9
Promedio de puntos de vida Ejército 2: 1.5
  
```

RANKING DE PODER EJÉRCITO PRIMERO (ORDENAMIENTO BURBURJA Y SELECCIÓN):

| | | |
|--|--|---|
|  | <p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |
| Aprobación: 2022/03/01 | Código: GUIA-PRLE-001 | Página: 10 |

Ranking de poder de Ejército 1 con Bubble Sort:

Soldado5X1 | Puntos de vida: 5
Soldado6X1 | Puntos de vida: 5
Soldado3X1 | Puntos de vida: 4
Soldado4X1 | Puntos de vida: 4
Soldado9X1 | Puntos de vida: 3
Soldado1X1 | Puntos de vida: 2
Soldado2X1 | Puntos de vida: 2
Soldado7X1 | Puntos de vida: 2
Soldado0X1 | Puntos de vida: 1
Soldado8X1 | Puntos de vida: 1

Ranking de poder de Ejército 1 con Selección:

Soldado5X1 | Puntos de vida: 5
Soldado6X1 | Puntos de vida: 5
Soldado3X1 | Puntos de vida: 4
Soldado4X1 | Puntos de vida: 4
Soldado9X1 | Puntos de vida: 3
Soldado1X1 | Puntos de vida: 2
Soldado2X1 | Puntos de vida: 2
Soldado7X1 | Puntos de vida: 2
Soldado0X1 | Puntos de vida: 1
Soldado8X1 | Puntos de vida: 1



RANKING DE PODER EJÉRCITO SEGUNDO (ORDENAMIENTO BURBURJA Y SELECCIÓN):

Ranking de poder de Ejército 2 con Bubble Sort:

Soldado1X2 | Puntos de vida: 2
Soldado0X2 | Puntos de vida: 1

Ranking de poder de Ejército 2 con Selección:

Soldado1X2 | Puntos de vida: 2
Soldado0X2 | Puntos de vida: 1

| | | |
|--|--|---|
|  | <p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación | | |
| Aprobación: 2022/03/01 | Código: GUIA-PRLE-001 | Página: 11 |

CONTABILIZACIÓN POR EJÉRCITO:

Total de puntos de vida Ejército 1: 29

Total de puntos de vida Ejército 2: 3

EJÉRCITO GANADOR BASADO EN EL CONTADOR DE PUNTOS DE VIDA:

☒ El Ejército 1 gana!

ITERADOR

☐ ¿Quieres crear nuevos ejércitos? (sí/no):



CONCLUSIONES

Se ha trabajado de acorde a la creación de 2 clases, atributos, el uso debido de HashMaps, métodos, según los requerimientos del ejercicios y solución adecuada. Se recrea un entorno HashMap para la resolución del problema dado.

METODOLOGÍA DE TRABAJO

Se colocó el avance inicial en github y la finalización del código donde se halla el código limpio y funcional. Se ha subido las pruebas en el link del repositorio del estudiante.

REFERENCIAS Y BIBLIOGRAFÍA

| | | |
|---|---|---|
|  | <p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p> |  |
| <p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p> | | |
| <p>Aprobación: 2022/03/01</p> | <p>Código: GUIA-PRLE-001</p> | <p>Página: 12</p> |

| Contenido y demostración | | Puntos | Checklis t | Estudiant e | Profeso r |
|--------------------------|--|--------|---------------|----------------|--------------|
| 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. Commits | Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación). | 4 | | 0 | |
| 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 | | 2 | |
| TOTAL | | 20 | | 14 | |

RUBRICA PARA EL CONTENIDO DEL INFORME Y DEMOSTRACIÓN