TECSUP

GUÍA DE LABORATORIO 4

“Java: Colecciones e hilos”

LABORATORIO

Objetivos:

- Implementar colecciones con Java.
- Escribir hilos.

Equipos, Materiales, Programas y Recursos:

- PC con Sistema Operativo con soporte a Java.
- IDE para Java

Introducción:

En esta sesión se detalla la utilización de Colecciones e Hilos.

Seguridad:

- Ubicar maletines y/o mochilas en el gabinete al final de aula de laboratorio.
- No ingresar con líquidos ni comida al aula de laboratorio.
- Al culminar la sesión de laboratorio, apagar correctamente la computadora y el monitor.

Preparación:

Durante el desarrollo de los temas de clase se tendrán ejercicios explicativos en cada uno de los puntos, ello le dará a la sesión una interacción de la teoría y la parte práctica, ya que en todo el momento el alumno podrá comprobar en su propia PC, todos los ítems del manual.

Procedimiento y Resultados:

COLECCIONES

Estudiante.java

```
package lab4;

public class Estudiante implements Comparable<Estudiante>  {

	private String nombres;
	private String apellidos;
	private String direccion;
	private int codMatricula;

	public int compareTo(Estudiante o) {
		String apellidos = o.getApellidos().toLowerCase();
		String apellidosLocal = this.getApellidos().toLowerCase();

		return apellidosLocal.compareTo(apellidos);
	}

// Crear su constructor y getters/setters por cada atributo

}
```

Colecciones1.java

```
public class Colecciones1 {

    public static void main(String args[]){

	Estudiante e1 = new Estudiante("Alberto", "Zapata", "Bolognesi 123",
100525);
	Estudiante e2 = new Estudiante("Benjamin", "Ayasta", "Libertad 987",
100526);
	Estudiante e3 = new Estudiante("Carlos", "Lopez", "Union 456",
100527);

	  // ArrayList
	ArrayList<Estudiante> c = new ArrayList<Estudiante>();
	c.add(e1);
	c.add(e2);
	c.add(e3);

	  Collections.sort(c);

	  for (Estudiante e : c) {
	      System.out.println(e.getNombres() + " " + e.getApellidos());
	  }
   }
}
```

Colecciones2.java

```
public class Colecciones2 {

    public static void main(String args[]){

      Estudiante e1 = new Estudiante("Alberto", "Zapata", "Bolognesi 123",
100525);
      Estudiante e2 = new Estudiante("Benjamin", "Ayasta", "Libertad 987",
100526);
      Estudiante e3 = new Estudiante("Carlos", "Lopez", "Union 456",
100527);

        // LinkedList
        LinkedList<Estudiante> pila = new LinkedList<Estudiante>();
        pila.addFirst(e1);
        pila.addFirst(e2);
        pila.addFirst(e3);

        ListIterator<Estudiante> ite = pila.listIterator();
        while(ite.hasNext()){
            Estudiante e = (Estudiante) pila.poll();
            System.out.println(e.getNombres());
        }
        System.out.println(pila.size());
  }
}
```

Colecciones3.java

```
public class Colecciones3 {

    public static void main(String args[]){

      Estudiante e1 = new Estudiante("Alberto", "Zapata", "Bolognesi 123",
100525);
      Estudiante e2 = new Estudiante("Benjamin", "Ayasta", "Libertad 987",
100526);
      Estudiante e3 = new Estudiante("Carlos", "Lopez", "Union 456",
100527);

        // HashSet
        HashSet<Estudiante> set = new HashSet<Estudiante>();
        set.add(e1);
        set.add(e2);
        set.add(e3);
        set.add(e1);

        for (Estudiante e : set) {
            System.out.println(e.getNombres() + " " + e.getApellidos());
        }
}
}
```

Colecciones4.java

```
public class Colecciones4 {

    public static void main(String args[]){

      Estudiante e1 = new Estudiante("Alberto", "Zapata", "Bolognesi 123",
100525);
      Estudiante e2 = new Estudiante("Benjamin", "Ayasta", "Libertad 987",
100526);
      Estudiante e3 = new Estudiante("Carlos", "Lopez", "Union 456",
100527);

        // HashMap
        HashMap<Integer, Estudiante> mapa = new HashMap<Integer,
Estudiante>();
        mapa.put(100525, e1);
        mapa.put(100526, e2);
        mapa.put(100527, e3);

        Estudiante e = mapa.get(100526);
        System.out.println(e.getNombres() );

        Collection<Estudiante> co = mapa.values();
        for (Estudiante es : co) {
            System.out.println(es.getNombres());
            }
    }
}
```

HILOS

- **Creación de un hilo por Herencia**

```
package laboratorio2;

public class HiloHerencia extends Thread {

	public void run() {
		for (int i = 0; i < 50; i++) {
			System.out.println(i);
		}
	}
}
```

```
package laboratorio2;

public class Test_HiloHerencia {

	public static void main(String[] args) {
		HiloHerencia h1 = new HiloHerencia();
		h1.start();
	}

}
```

- **Creación de un hilo por Implementación**

```
package laboratorio2;

public class HiloInterfase implements Runnable {

	public void run() {
		for (int i = 0; i < 50; i++) {
			System.out.println(i);
		}
	}
}
```

```
package laboratorio2;

public class Test_HiloInterfase {

	public static void main(String[] args) {
		HiloInterfase st = new HiloInterfase ();
		Thread th = new Thread(st);
		th.start();
	}
}
```

- **Método sleep del Thread**

```
package laboratorio2;

public class SimpleDelayThread extends Thread {

	public void run() {
		//
		for (int i = 0; i < 5; i++) {
			try {
				// 1000 ms
				Thread.sleep(1000);
			} catch (InterruptedException e) {
				e.printStackTrace();
			}
			System.out.println(i);
		}
	}
}
```

```
package laboratorio2;

public class Test_SimpleDelayThread {

	public static void main(String[] args) {
		SimpleDelayThread sd = new SimpleDelayThread();
		sd.start();
	}
}
```

- **Generando retardos aleatorios con Hilos**

```
package laboratorio2;

public class Simple2DelayThread extends Thread {

	public String name = null;
	public int delay = 0;

	public Simple2DelayThread(String str, int d) {
		this.name = str;
		this.delay = d;
	}

	public void run() {
		try {
			Thread.sleep(delay);
		} catch (InterruptedException e) {
			e.printStackTrace();
		}
		System.out.println("name : " + name + " | delay : " + delay);
	}
}
```

```
package laboratorio2;

public class Test_Simple2DelayThread {

	public static void main(String[] args) {

		int delay1 = (int)(Math.random()*2000);
		int delay2 = (int)(Math.random()*2000);
		int delay3 = (int)(Math.random()*2000);

		Simple2DelayThread thread1 = new Simple2DelayThread("thread1",delay1);
		Simple2DelayThread thread2 = new Simple2DelayThread("thread2",delay2);
		Simple2DelayThread thread3 = new Simple2DelayThread("thread3",delay3);

		thread1.start();
		thread2.start();
		thread3.start();
	}
}
```

- **Manejando prioridades con Hilos**

```
package laboratorio2;

public class Prioridades {

	public static void main(String[] args) {

		Simple2DelayThread thread1 = new Simple2DelayThread("thread1", 0);
		Simple2DelayThread thread2 = new Simple2DelayThread("thread2", 0);

		// Modificando las prioridades
		thread1.setPriority(Thread.MIN_PRIORITY);
		thread2.setPriority(Thread.MAX_PRIORITY);

		thread1.start();
		thread2.start();
	}
}
```

Ejercicios

- Escribir un programa que cada 5 segundos muestre en la consola la memoria RAM disponible para el JVM.

```
long mem0 = Runtime.getRuntime().totalMemory();
long mem1 = Runtime.getRuntime().freeMemory();
long mem2 = Runtime.getRuntime().availableProcessors();
long mem3 = Runtime.getRuntime().maxMemory();
```

- Escribir un programa que cada 10 segundos muestre en consola la información de los procesos que está corriendo un usuario del sistema. Ejemplo de código:

```
try {
Process proceso = Runtime.getRuntime().exec("ps -ef");

BufferedReader stdInput = new BufferedReader(new InputStreamReader(
                                        proceso.getInputStream()));

BufferedReader stdError = new BufferedReader(new InputStreamReader(
                                        proceso.getErrorStream()));


        String s = null;

        System.out.println("Here is the standard output of the command:\n");
        while ((s = stdInput.readLine()) != null) {
                System.out.println(s);
        }
        System.out.println("Here is the standard error of the command (if any):\n");

        while ((s = stdError.readLine()) != null) {
             System.out.println(s);
        }

        System.exit(0);

} catch (IOException e) {
        e.printStackTrace();
}
```

Conclusiones:

En la presente sesión, se detalló el uso de las Colecciones e Hilos.