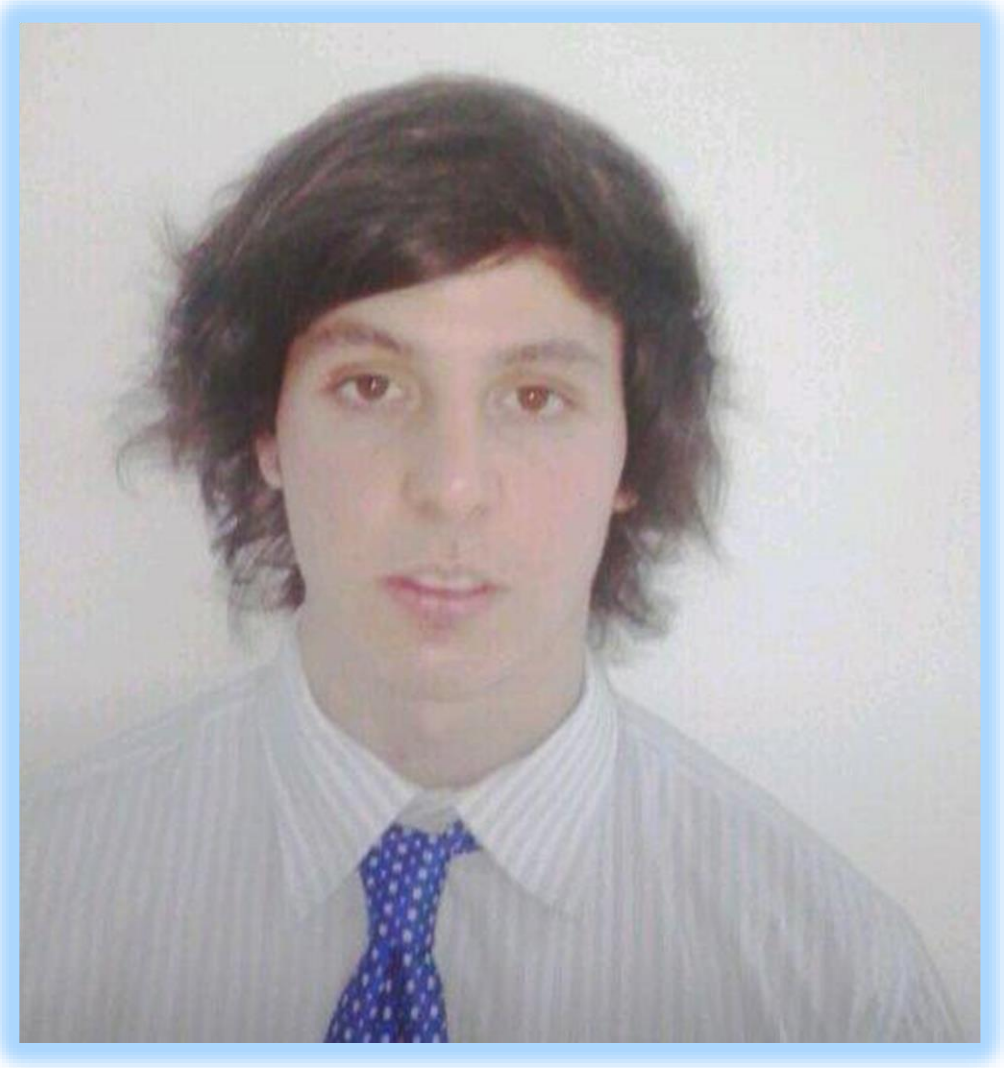
Universidad ORT.

Facultad de ingeniería

Alumno: Joaquín Ruiz Puppo



Nro estudiante: 206164

Carpeta de documentación del segundo obligatorio.

Índice:

Datos de prueba:

En menú (se muestran 9 opciones) :	Lo que se espera:	Resultado:
Ingreso 10	“Elija una opción válida (1-9)”	Ok
Ingreso 0	“Elija una opción válida (1-9)”	Ok
Ingreso Er	“Ingrese solo números”	
No ingreso nada (oprimo enter)	No me acepta el dato hasta que ingrese algo.	Ok
Ingreso 1	Entra a la opción de registro de inspector.	Ok
Ingreso 2	Entra a la opción de registro de encargado	
Ingreso 3	1-Si no hay ningún encargado registrado se muestra. ” No se puede ingresar a esta opción no hay ningún encargado/a registrado/a”. 2-Entra a la opción de registro de actividad.	1-ok 2-ok
Ingreso 4	1-Si no hay ningún inspector o actividad registrado se muestra “No se puede ingresar a esta opción por falta de datos, no hay ningún inspector/a registrado/a o ninguna actividad registrada.” 2-Entra a la opción de registro de inspección.	1-ok 2-ok
Ingreso 5	1-Si no hay ninguna inspección registrada se muestra	1-ok 2-ok

	"No hay ninguna inspección registrada, no se puede ejecutar la opción seleccionada." 2-Entra a la opción 5.	
Ingreso 6	1-Si no hay ninguna inspección registrada se muestra " No hay ninguna inspección registrada, no se puede ejecutar la opción seleccionada." 2-Entra a la opción 6.	1-ok 2-ok
Ingreso 7	1-Si no hay ninguna inspección registrada se muestra " No hay ninguna inspección registrada, no se puede ejecutar la opción seleccionada." 2-Entra a la opción 7.	1-ok 2-ok
Ingreso 8	1-Si no hay ninguna inspección registrada se muestra " No hay ninguna inspección registrada, no se puede ejecutar la opción seleccionada." 2-Entra a la opción 8.	1-oK 2-oK
Ingreso 9	Termina el programa	Ok
En la opción 1 (Registro inspector)	Lo que se espera	Resultado
1-Ingresando el nombre del inspector, ingreso 23. • 1.1-Ingreso ",.". • 1.2-Oprimo enter sin ingresar nada. • 1.3-Ingreso espacios vacíos. • 1.4 Ingreso "Joaquín Ruiz" 2-Ingresando la cedula: Ingreso abcdefg. • 2.1-Ingreso 123456. • 2.2-Ingreso 123456789 • 2.3-Ingreso 12345678.	1- "Ingrese el dato ingresando únicamente letras, sin ningún carácter especial" • 1.1-Se muestra lo mismo que arriba. • 1-2-Se muestra lo mismo que arriba. • 1-3-Semuestra lo mismo que arriba. • 1.4-Toma el dato salta al siguiente paso. 2-"Ingrese una cedula valida." • 21-Se muestra lo mismo que arriba	1- • 1-ok • 1.1-ok • 1.2-ok • 1.3-ok • 1.4-ok 2-ok • 2.1-ok • 2.2-ok • 2.3-ok 3-ok • 3.1-ok

3-Ingreso ade. 3.1-Ingreso 20.	• 2.2-Semuestra lo mismo que arriba. • 2.3-Toma el dato y pasa al siguiente paso. 3-“Ingrese solamente números” • 3.1-Toma la edad y registra al inspector.	
En la opción 2 (Registro encargado)	Lo que se espera	Resultado
1-Ingresando el nombre de encargado. 2-Ingresando la cedula del encargado 3-Ingresando la dirección del encargado 3.-Ingreso ¡!,. 3.1-Ingreso Tomas DIago 874	1-Lo mismo que en registro de inspector. 2-Lo mismo que en registro de inspector. 3-Se muestra “Ingrese una opción válida”. 3.1-Toma el dato y queda completado el registro de encargado.	1-ok 2-ok 3-ok • 3.1-ok
En la opción 3 (Registro de actividad)	Lo que se espera	Resultado
1-Ingresando la sección dela actividad, ingreso 11. • 1.1-Ingreso 0. • 1.2-Ingresando la descripción de la actividad ingreso.	1-“Ingrese una sección valida (1-10)”. 1.1-Lo mismo que arriba. 1.2- Se ejecuta las validaciones de letras ya mencionadas anteriormente.	1-ok • 1.1-ok • 1.2-ok
En la opción 4	Lo que se espera	Resultado
1-Eligiendo el inspector de la lista de inspectores, me paso de rango. • 1.1-Elijo un inspector dentro del rango. • 1.2-Eligiendo la actividad de la lista de actividades me paso de rango. • 1.3-Elijo una actividad dentro del rango. • 1.4-Ingresando el día de la inspección. • 1.5-Ingresando el mes de la inspección.	1- “Ingrese una opción valida (1-rango de la lista). • 1.1-Toma el dato y pasa al siguiente paso. • 1.2- Ingrese una opción válida (1-rango de la lista). • 1.3-toma y pasa al siguiente dato. • 1.4- (A partir de este punto se repiten las validaciones ya mencionadas previamente. De ingreso de números, letras, rangos etc.)	1-ok • 1.1-ok • 1-2-ok • 1.3-ok • (1.4-1.9)-ok

• 1.6-Ingresando comentarios de la inspección. • 1.7-Ingresando cuanto duro la inspección. • 1.8-Ingresando el riesgo a evaluar. • 1.9-Ingresando el resultado de la inspección.		
En la opción 5 (Mostrar inspecciones)	Lo que se espera	Resultado
Ingreso a la opción	Muestra las opciones ordenadas por fecha.	Ok
En la opción 6	Lo que se espera	Resultado
1-Ingreso el mes 100 1.1-Ingreso 0 1.2-Ingreso 6.	1-Ingrese un mes valido. 1.1-Lo mismo. 1.2-Si hay una inspección no aprobada en ese mes la muestra.	1-ok • 1.1-ok • 1.2-ok
En la opción 7	Lo que se espera	Resultado
Entro a la opción	1-Si no hay ninguna inspección registrada “No hay ninguna inspección registrada, no se puede ejecutar la opción seleccionada.” 1.1-Muestra la sección con máximas inspecciones realizadas y las inspecciones aprobadas y no aprobadas por sección.	1-ok • 1.1-ok
En la opción 8	Lo que se espera	Resultado
1-Elijo inspector a dar de baja y no tiene inspecciones realizadas. 2-Elijo un inspector que tiene al menos una inspección realizada.	1- “El inspector (nombre del inspector elegido) fue dado de baja exitosamente.” 2- “El inspector (nombre del inspector elegido) no puede ser dado de baja porque tiene al menos una inspección hecha.”	1-ok 2-ok
En la opción 9	Lo que se espera	Resultado
Entra a opción	Termina el programa	Ok

Clases

Sistema:

```
public class Sistema {
    private  ArrayList<Actividad> actividades = new ArrayList<Actividad>();
    private  ArrayList<Inspector> inspectores = new ArrayList<Inspector>();
    private  ArrayList<Encargado> encargados = new ArrayList<Encargado>();
    private  ArrayList<Inspeccion> inspecciones = new ArrayList<Inspeccion>();



    public Sistema() {
        actividades = new ArrayList<Actividad>();
        inspectores = new ArrayList<Inspector>();
        encargados = new ArrayList<Encargado>();
        inspecciones = new ArrayList<Inspeccion>();




    }

    public void setActividades(ArrayList<Actividad> actividades) {
        this.actividades = actividades;
    }

    public void setInspectores(ArrayList<Inspector> inspectores) {
        this.inspectores = inspectores;
    }
```

```
    public void setEncargados(ArrayList<Encargado> encargados) {
        this.encargados = encargados;
    }

    public void setInspecciones(ArrayList<Inspeccion> inspecciones) {
        this.inspecciones = inspecciones;
    }



    public  ArrayList<Actividad> getActividades() {
        return actividades;
    }

    public  ArrayList<Inspector> getInspectores() {
        return inspectores;
    }

    public  ArrayList<Encargado> getEncargados() {
        return encargados;
    }

    public  ArrayList<Inspeccion> getInspecciones() {
        return inspecciones;
    }


}
```

Inspector:

```
package obligatorio2;


public class Inspector extends Persona{
    private int edad;

    public Inspector( String nombre , String cedula ,int edad) {
        super (nombre , cedula);
        this.setEdad(edad);

    }
    public Inspector(){
        this.setEdad(0);
    }


    public int getEdad() {
        return edad;
    }

    public void setEdad(int edad) {
        this.edad = edad;
    }
    @Override
    public String toString(){
        return "-------Inspector-------\nNombre : "+this.getNombre()+"\nCedula :
"+this.getCedula()+"\nEdad : "+this.getEdad()+"\n----------------------";
    }
```

```
}
```

Encargado:

```
package obligatorio2;


public class Encargado extends Persona{
   private String direccion;
   public Encargado(){
      this.setDireccion("Sin nombre");

   }

   public Encargado(String nombre, String cedula, String direccion) {
      super (nombre , cedula);
      this.direccion = direccion;
   }




   public String getDireccion() {
      return direccion;
   }

   public void setDireccion(String direccion) {
      this.direccion = direccion;
   }
   @Override
```

```
    public String toString(){
        return "-----Encargado-----\nNombre : "+this.getNombre()+"\nCedula :
"+this.getCedula()+"\nDireccion : "+this.getDireccion()+"\n------------------";

    }

}
```

Persona:

```
package obligatorio2;



public class Persona {

    private String nombre;
    private String cedula;
    public Persona(){
        this.setNombre("Sin nombre");
        this.setCedula("Sin cedula");
    }


    public Persona(String nombre, String cedula) {
        this.nombre = nombre;
        this.cedula = cedula;
    }


    public String getNombre() {
        return nombre;
```

```
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getCedula() {
        return cedula;
    }

    public void setCedula(String cedula) {
        this.cedula = cedula;
    }
    @Override
    public String toString(){
        return "Nombre : "+this.getNombre()+"\nCedula : "+this.getCedula();
    }
    @Override
    public boolean equals(Object o){
        Persona p = (Persona) o;
        return this.getCedula().equals(p.getCedula());

    }


}
```

Inspeccion:

```
package obligatorio2;
```

```
public class Inspeccion implements Comparable<Inspeccion>{
  private Inspector inspector;
  private Actividad actividad;
  private int dia;
  private int mes;
  private int horaReal;
  private String comentario;
  private String resultado;
  private String riesgoo;
  private String month;

   public Inspeccion(Inspector inspector, Actividad actividad, int dia, int mes, int
horaReal, String comentario, String resultado, String riesgoo, String month) {
      this.inspector = inspector;
      this.actividad = actividad;
      this.dia = dia;
      this.mes = mes;
      this.horaReal = horaReal;
      this.comentario = comentario;
      this.resultado = resultado;
      this.riesgoo = riesgoo;
      this.month = month;
   }


   public Inspector getInspector() {
      return inspector;
   }
```

```
public void setInspector(Inspector inspector) {
    this.inspector = inspector;
}

public Actividad getActividad() {
    return actividad;
}

public void setActividad(Actividad actividad) {
    this.actividad = actividad;
}

public int getDia() {
    return dia;
}

public void setDia(int dia) {
    this.dia = dia;
}

public int getMes() {
    return mes;
}

public void setMes(int mes) {
    this.mes = mes;
}
```

```
public int getHoraReal() {
    return horaReal;
}

public void setHoraReal(int horaReal) {
    this.horaReal = horaReal;
}

public String getComentario() {
    return comentario;
}

public void setComentario(String comentario) {
    this.comentario = comentario;
}

public String getResultado() {
    return resultado;
}

public void setResultado(String resultado) {
    this.resultado = resultado;
}

public String getRiesgoo() {
    return riesgoo;
}

public void setRiesgoo(String riesgoo) {
```

```
        this.riesgoo = riesgoo;
    }

    public String getMonth() {
        return month;
    }

    public void setMonth(String month) {
        this.month = month;
    }
    @Override
    public String toString(){
        return "Inspeccion\n----------\nInspector :
"+inspector.getNombre()+"\nActividad : "+actividad.getDescripcion()+"\nDia :
"+this.getDia()+"\nMes : "+this.getMonth()+"\nComentarios :
"+this.getComentario()+"\nHoras reales de la actividad :
"+this.getHoraReal()+"\nRiesgo evaluado : "+this.getRiesgoo()+"\nResultado :
"+this.getResultado()+"\n----------\n";

    }
    @Override
    public int compareTo(Inspeccion i){
        int diferencia = 0;
        diferencia = this.getMes()-i.getMes();
        if(diferencia == 0){
            diferencia = this.getDia()-i.getDia();

        }
        return diferencia;
    }
```

}

Actividad:

```
package obligatorio2;


public class Actividad {


    private int seccion;

    private String descripcion;

    private int duracionHoras;

    private int RiesgoPrincipal;

    private int RiesgoSecundario;

    private String riesgo1;

    private String riesgo2;

    private Encargado encargado;


    public Actividad(int seccion, String descripcion, int duracionHoras, int
RiesgoPrincipal, int RiesgoSecundario, String riesgo1, String riesgo2, Encargado
encargado) {

        this.seccion = seccion;

        this.descripcion = descripcion;

        this.duracionHoras = duracionHoras;

        this.RiesgoPrincipal = RiesgoPrincipal;

        this.RiesgoSecundario = RiesgoSecundario;

        this.riesgo1 = riesgo1;

        this.riesgo2 = riesgo2;

        this.encargado = encargado;

    }
```

```
public Encargado getEncargado() {
    return encargado;
}

public void setEncargado(Encargado encargado) {
    this.encargado = encargado;
}

public Actividad() {
    this.setDescripcion("Sin descripcion");
    this.setDuracionHoras(0);
    this.setRiesgo1("Sin riesgo principal");
    this.setRiesgo2("Sin riesgo secundario");
    this.setRiesgoPrincipal(0);
    this.setRiesgoSecundario(0);
    this.setSeccion(0);
    this.setEncargado(encargado);

}

public int getSeccion() {
    return seccion;
}

public void setSeccion(int seccion) {
    this.seccion = seccion;
}
```

```
public String getDescripcion() {
    return descripcion;
}

public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
}

public int getDuracionHoras() {
    return duracionHoras;
}

public void setDuracionHoras(int duracionHoras) {
    this.duracionHoras = duracionHoras;
}

public int getRiesgoPrincipal() {
    return RiesgoPrincipal;
}

public void setRiesgoPrincipal(int RiesgoPrincipal) {
    this.RiesgoPrincipal = RiesgoPrincipal;
}

public int getRiesgoSecundario() {
    return RiesgoSecundario;
}

public void setRiesgoSecundario(int RiesgoSecundario) {
```

```
        this.RiesgoSecundario = RiesgoSecundario;
    }

    public String getRiesgo1() {
        return riesgo1;
    }

    public void setRiesgo1(String riesgo1) {
        this.riesgo1 = riesgo1;
    }

    public String getRiesgo2() {
        return riesgo2;
    }

    public void setRiesgo2(String riesgo2) {
        this.riesgo2 = riesgo2;
    }

    @Override
    public String toString() {
        return "-------Actividad-------\nDescripcion : " + this.getDescripcion() +
"\nEncargado : " + encargado.getNombre() + "\nSeccion : " + this.getSeccion() +
"\nDuracion en horas : " + this.getDuracionHoras() + "\nRiesgo principal : " +
this.getRiesgo1() + "\nRiesgo secundario : " + this.getRiesgo2() + "\n----------------------";
    }

}
```

Consola:

```
package obligatorio2;
```

```
import java.util.*;

public class Consola {

   static Scanner in = new Scanner(System.in);

   private static void ShowActivities(ArrayList<Actividad> actividades) {
      for (int i = 0; i < actividades.size(); i++) {
         System.out.println((i + 1) + ")" + "\n" + actividades.get(i));

      }
   }

   private static void NonApprovedInspections(ArrayList<Inspeccion> inspecciones) {
      int mes = 0;
      int contador = 0;
      int numerador=1;
      System.out.println("Ingrese un mes : ");
      mes = ValidMonth();
      for (int i = 0; i < inspecciones.size(); i++) {
         if ((inspecciones.get(i).getResultado().equalsIgnoreCase("no aprobado")) &&
(inspecciones.get(i).getMes() == mes)) {
            contador++;
            System.out.println("Inspecciones no aprobadas en
"+inspecciones.get(i).getMonth()+" :\n"+(numerador)+")\n"+inspecciones.get(i));
            numerador++;

         }
```

```
        }
        if(contador==0){
            System.out.println("No hay ninguna inspeccion no aprobada en el mes
seleccionado.");
        }




    }

    private static Encargado ElegirEncargado(ArrayList<Encargado> encargados) {
        int opcion = 0;
        int numerador = 1;
        boolean ok = false;
        Encargado encargadox = new Encargado();
        System.out.println("Elija un encargado :");
        for (int i = 0; i < encargados.size(); i++) {
            System.out.println(numerador + ")- " + encargados.get(i).getNombre());
            numerador++;

        }
        opcion = Numbers();
        while (!ok) {
            if ((opcion < 1) || (opcion >= numerador)) {
```

```
            System.out.println("Ingrese una opcion valida (1-" + (numerador - 1) + ")");
            opcion = Numbers();
        } else {
            ok = true;
        }

    }

    numerador = opcion;
    encargadox = encargados.get((opcion - 1));

    return encargadox;

  }

  private static void ShowInspections(ArrayList<Inspeccion> inspecciones) {
    Collections.sort(inspecciones);
    for (int i = 0; i <inspecciones.size(); i++) {
        System.out.println("_____Inspeccion
"+(i+1)+"_____\n"+inspecciones.get(i));

    }


    }



  private static void RegisterActivity(ArrayList<Encargado> encargados,
ArrayList<Actividad> actividades) {
```

```
        int opcion = 0;
        int seccion = 0;
        String descripcion = "";
        int duracionHoras = 0;
        int RiesgoPrincipal = 0;
        int RiesgoSecundario = 0;
        String riesgo1 = "";
        String riesgo2 = "";
        Encargado jefe = new Encargado();
        boolean ok = false;

        do {
            int numerador = 0;
            System.out.println("_____Registro de actividad _____\n");
            System.out.println("Ingrese la seccion de la actividad :");
            seccion = ValidSection();
            in.nextLine();
            System.out.println("Ingrese la descripcion de la actividad :");
            descripcion = Letters();
            System.out.println("Ingrese jefe a cargo de la actividad (Encargado de la actividad) ");
            jefe = ElegirEncargado(encargados);
            System.out.println("Encargado de la actividad : " + jefe.getNombre());
            System.out.println("Ingrese la duracion de la actividad en horas :");
            duracionHoras = ValidHours();
            in.nextLine();
            System.out.println("Ingrese el riesgo principal de la actividad \n1-Riesgo fisico\n2-Riesgo quimico\n3-Riesgo biologico\n4-Riesgo sicosociales");
            RiesgoPrincipal = ValidRisk();
```

```
        in.nextLine();
        System.out.println("Ingrese el riesgo secundario , sin ser el ya elegido como
principal (" + RiesgoPrincipal + ")");
        RiesgoSecundario = SameRisk(RiesgoPrincipal);
        in.nextLine();
        switch (RiesgoPrincipal) {
           case 1:
              riesgo1 = "Riesgo Fisico";
              break;
           case 2:
              riesgo1 = "Riesgo Quimico";
              break;
           case 3:
              riesgo1 = "Riesgo Biologico";
              break;
           case 4:
              riesgo1 = "Riesgo Sicosociales";
              break;
           default:
              break;

        }
        switch (RiesgoSecundario) {
           case 1:
              riesgo2 = "Riesgo Fisico";
              break;
           case 2:
              riesgo2 = "Riesgo Quimico";
              break;
           case 3:
```

```
                riesgo2 = "Riesgo Biologico";
                break;
            case 4:
                riesgo2 = "Riesgo Sicosociales";
                break;
            default:
                break;

        }
        Actividad aActivity = new Actividad(seccion, descripcion, duracionHoras,
RiesgoPrincipal, RiesgoSecundario, riesgo1, riesgo2, jefe);
        actividades.add(aActivity);
        System.out.println("La actividad fue registrada con exito!\n" + aActivity +
"\n_____
_____");
        System.out.println("Desea registrar otra actividad ?\n1)-Si\n2)-No");
        opcion = Numbers();
        while (!ok) {
            if (opcion < 1 || opcion > 2) {
                System.out.println("Ingrese una opcion valida ( 1-2 ) :");
                opcion = Numbers();
            } else {
                ok = true;
            }
        }
        in.nextLine();
    } while (opcion != 2);

}
```

```
private static int SameRisk(int riesgoP) {
    int risky = 0;
    boolean ok = false;
    risky = ValidRisk();
    while (!ok) {
        if ((risky) == (riesgoP)) {
            System.out.println("Ingrese un riesgo que no coincida con el principal :");
            risky = ValidRisk();
        } else {
            ok = true;
        }
    }
    return risky;
}

private static int ValidRisk() {
    int risk = 0;
    boolean ok = false;
    risk = Numbers();
    while (!ok) {
        if ((risk < 1) || (risk > 4)) {
            System.out.println("Ingrese un riesgo valido (1-4)");
            risk = Numbers();
        } else {
            ok = true;
        }
    }
    return risk;
}
```

```
private static int ValidHours() {
    int hours = 0;
    boolean ok = false;
    hours = Numbers();
    while (!ok) {
        if ((hours < 1)) {
            System.out.println("Ingrese una cantidad de horas valida ( No negativa )");
            hours = Numbers();

        } else {
            ok = true;
        }

    }
    return hours;

}

private static int ValidSection() {
    int section = 0;
    boolean ok = false;
    section = Numbers();
    while (!ok) {
        if ((section <= 0) || (section > 10)) {
            System.out.println("Ingrese una seccion valida (1-10) :");
            section = Numbers();
        } else {
            ok = true;
```

```
        }
      }
      return section;
   }

   private static int Numbers() {
      int number = 0;
      boolean ok = false;
      while (!ok) {
         try {
            number = in.nextInt();
            ok = true;
         } catch (Exception e) {
            System.out.println("Ingrese solamente numeros :");
            in.next();
         }
      }
      return number;
   }

   private static String Letters() {
      String letter = "";
      boolean ok = false;
      letter = in.nextLine();
      letter = letter.trim();
      while (!ok) {
         if ((!letter.matches("[A-Za-z ]*")) || (letter.isEmpty())) {
            System.out.println("Ingrese el dato ingresando unicamente letras , sin nigun
caracter especial :");
            letter = in.nextLine();
```

```
            letter = letter.trim();

        } else {
            ok = true;
        }

    }
    return letter;
  }

  private static String Id() {
    String id = "";
    boolean ok = false;
    id = in.nextLine();
    id = id.replace("-", "");
    id = id.replace(" ", "");
    id = id.trim();
    while (!ok) {
      if ((!id.matches("[0-9]*")) || (id.isEmpty()) || (id.length() < 7) || (id.length() > 8))
{
        System.out.println("Ingrese una cedula valida :");
        id = in.nextLine();
        id = id.replace("-", "");
        id = id.replace(" ", "");
        id = id.trim();
      } else {
        ok = true;
      }

    }
```

```
        return id;


    }


    private static void RegisterInspector(ArrayList<Inspector> inspectores ,
ArrayList<Encargado> encargados) {
        String cedula = "";
        String nombre = "";
        int edad = 0;
        int opcion = 0;
        boolean esta = false;
        boolean ok = false;
        do {
            System.out.println("_____Registro de
inspector_____\n");
            System.out.println("Ingrese el nombre del inspector :");
            nombre = Letters();
            System.out.println("Ingrese la cedula del inspector :");
            cedula = Id();
            System.out.println("Ingrese la edad del inspector :");
            edad = Numbers();
            while(edad<20){
                System.out.println("Ingrese una edad valida :");
                edad = Numbers();
            }
            Inspector aInspector = new Inspector(nombre, cedula, edad);
            if(inspectores.contains(aInspector) || encargados.contains(aInspector)){
                esta = true;
                if(esta  && inspectores.contains(aInspector)){
```

```
            System.out.println("El inspector "+aInspector.getNombre()+" ya esta
registrado en el sistema.");
            opcion = 2;
        }
        else if(esta && encargados.contains(aInspector)){
            System.out.println(aInspector.getNombre()+" ya esta registrado/a como
encargado , por lo tanto no puede ser registrado/a como inspector.");
            opcion = 2;
        }




    }else{
        inspectores.add(aInspector);
    System.out.println("El inspector fue registrado con exito !\n" + aInspector +
"\n_____
_____");
    System.out.println("Desea registrar otro inspector?\n1-Si\n2-No");
    opcion = Numbers();
    while (!ok) {
        if (opcion < 1 || opcion > 2) {
            System.out.println("Ingrese una opcion valida ( 1-2) :");
            opcion = Numbers();
        } else {
            ok = true;
        }


    }
```

```
        }
        in.nextLine();

    } while (opcion != 2);

}

private static void SearchById(ArrayList<Inspector> inspectores) {
    String ci = "";
    System.out.println("Ingrese la cedula del inspector :");
    ci = Id();
    for (int i = 0; i < inspectores.size(); i++) {
        if ((inspectores.get(i).getCedula().equals(ci))) {
            System.out.println("Inspector : \n" + inspectores.get(i).getNombre() + "\n");

        }

    }

}

private static void backToMenu() {
    String enter = "";
    boolean ok = false;
    System.out.println("Volver al menu : Enter");
    enter = in.nextLine();
    while (!ok) {
        if (enter.isEmpty()) {
            ok = true;
```

```
            } else {
                System.out.println("Volver al menu : enter");
                enter = in.nextLine();
            }
        }

    }

    private static String Adress() {
        String letterN = "";
        boolean ok = false;
        letterN = in.nextLine();
        while (!ok) {
            if (letterN.matches("[A-Za-z0-9 ]*")) {
                ok = true;
            } else {
                System.out.println("Ingrese una direccion valida :");
                letterN = in.nextLine();

            }
        }
        return letterN;
    }

    private static void RegistrarEncargado(ArrayList<Encargado> encargados ,
ArrayList<Inspector> inspectores) {
        int opcion = 0;
        boolean esta = false;
        boolean ok = false;
        do {
```

```
        String name = "";
        String adress = "";
        String id = "";
        System.out.println("_____Registro de
encargado_____\n");
        System.out.println("Ingrese el nombre del encargado : ");
        name = Letters();
        System.out.println("Ingrese la cedula del encargado : ");
        id = Id();
        System.out.println("Ingrese la direccion del encargado :");
        adress = Adress();
        Encargado unEncargado = new Encargado(name, id, adress);
        if(encargados.contains(unEncargado) || inspectores.contains(unEncargado)){
           esta = true;
           if(esta && encargados.contains(unEncargado)){
              System.out.println("El encargado "+unEncargado.getNombre()+" ya esta
registrado/a en el sistema.");
              opcion=2;
           }
           else if(esta && inspectores.contains(unEncargado)){
              System.out.println(unEncargado.getNombre()+" ya esta registrado/a como
inspector , por lo tanto no puede ser registrado/a como encargado.");
              opcion = 2;

           }

        }else{
           System.out.println("El encargado fue registrado con exito !\n" + unEncargado
+
```

```
"\n_____
_____");
            encargados.add(unEncargado);
            System.out.println("Desea registrar otro encargado ?\n1-Si\n2-No");
            opcion = Numbers();
            while (!ok) {
                if (opcion < 1 || opcion > 2) {
                    System.out.println("Ingrese una opcion valida ( 1-2) :");
                    opcion = Numbers();
                } else {
                    ok = true;
                }
            }
             in.nextLine();


            }




        } while (opcion != 2);


    }

    private static int ValidMonth() {
        int mes = 0;
        boolean ok = false;
        mes = Numbers();
        while (!ok) {
            if ((mes < 1) || (mes > 12)) {
                System.out.println("Ingrese un mes valido (1-12) :");
```

```
            mes = Numbers();
        } else {
            ok = true;
        }
    }
    return mes;
}

public static Inspector ChooseInspector(ArrayList<Inspector> inspectores) {
    //resto (-1) al numerador porque arranca en 1 entonces queda con una unaidad mas que los elentos d ela lista.
    int opcion = 0;
    int numerador = 1;
    boolean ok = false;
    Inspector inspectorx = new Inspector();
    System.out.println("Elija un inspector : ");
    for (int i = 0; i < inspectores.size(); i++) {
        System.out.println(numerador + ")-" + inspectores.get(i).getNombre());
        numerador++;

    }
    opcion = Numbers();
    while (!ok) {
        if ((opcion < 1) || (opcion >= numerador)) {
            System.out.println("Ingrese una opcion valida (1-" + (numerador - 1) + ")");
            opcion = Numbers();

        } else {
            ok = true;
```

```
            }
        }

        numerador = opcion;
        inspectorx = inspectores.get((opcion - 1));

        return inspectorx;
    }

    private static Actividad ChooseActivity(ArrayList<Actividad> actividades) {
        // resto al numerador (-1) porque lo arranque en 1 entonces el for al sumarlo 1 cada pasada queda con uno de mas a la cantidad de actividades en la lista.
        int opcion = 0;
        int numerador = 1;
        boolean ok = false;
        Actividad actividadx = new Actividad();
        for (int i = 0; i < actividades.size(); i++) {
            System.out.println(numerador + ")- " + actividades.get(i).getDescripcion());
            numerador++;

        }

        opcion = Numbers();
        while (!ok) {
            if ((opcion < 1) || (opcion >= numerador)) {
                System.out.println("Ingrese una opcion valida (1-" + (numerador - 1 + ")"));
                opcion = Numbers();

            } else {
                ok = true;
```

```
        }
    }
    numerador = opcion;
    actividadx = actividades.get((opcion - 1));

    return actividadx;

}

private static int ValidDays() {
    int dia = 0;
    boolean ok = false;
    dia = Numbers();
    while (!ok) {
        if ((dia < 1) || (dia > 30)) {
            System.out.println("Ingrese un dia valido (1-30)");
            dia = Numbers();
        } else {
            ok = true;
        }
    }
    return dia;

}

private static int Option1or2() {
    int option = 0;
    boolean ok = false;
```

```
        option = Numbers();
        while (!ok) {
            if ((option < 1) || (option > 2)) {
                System.out.println("Ingrese una opcion valida (1-2)");
                option = Numbers();
            } else {
                ok = true;
            }
        }
        return option;
    }

    private static void RegisterInspection(ArrayList<Inspector> inspectores,
ArrayList<Actividad> actividades, ArrayList<Inspeccion> inspecciones) {
        int mes = 0;
        String month = "";
        int dia = 0;
        int riesgo = 0;
        int result = 0;
        String resultado = "";
        int horaReal = 0;
        String comentario = "";
        Inspector inspector = new Inspector();
        Actividad actividad = new Actividad();
        String riesgoo = "";
        int opcion = 0;
        int No = 0;
        boolean ok = false;
        do {
```

```
        System.out.println("_____Registro de
Inspeccion_____\n");
        System.out.println("Que inspector realiza la inspeccion ?");
        inspector = ChooseInspector(inspectores);
        System.out.println("Inspector que realiza la actividad :\n" + inspector);
        System.out.println("Que actividad se va a evaluar ?");
        actividad = ChooseActivity(actividades);

        System.out.println("Actividad a ser evaluada : \n" + actividad);
        System.out.println("Ingrese el dia que se realiza la inspeccion : ( 1-30)");
        dia = ValidDays();
        in.nextLine();
        System.out.println("Ingrese el mes de la inspeccion : ");
        mes = ValidMonth();
        switch (mes) {
           case 1:
              month = "Enero";
              break;
           case 2:

              month = "Febrero";
              break;
           case 3:
              month = "Marzo";
              break;
           case 4:
              month = "Abril";
              break;
           case 5:
              month = "Mayo ";
```

```
            break;
        case 6:
            month = "Junio";
            break;
        case 7:
            month = "Julio";
            break;
        case 8:
            month = "Agosto";
            break;
        case 9:
            month = "Setiembre";
            break;
        case 10:
            month = " Octubre ";
            break;
        case 11:
            month = " Noviembre";
            break;
        case 12:
            month = "Diciembre";
            break;
        default:
            break;

    }
    in.nextLine();
    System.out.println("Ingrese los comentarios realizados por el inspector :");
    comentario = Letters();
```

```
        System.out.println("Ingrese cuanto duro realmente la actividad  :");
        horaReal = ValidHours();
        in.nextLine();
        System.out.println("Ingrese el riesgo a evaluar :\n1-" + actividad.getRiesgo1() +
"\n2-" + actividad.getRiesgo2());
        opcion = Option1or2();
        in.nextLine();
        if (opcion == 1) {
           riesgoo = actividad.getRiesgo1();

        } else if (opcion == 2) {
           riesgoo = actividad.getRiesgo2();
        }

        System.out.println("Ingrese el resultado de la inspeccion\n1-Aprobado\n2-No
aprobado");
        result = Option1or2();
        in.nextLine();
        if ((result == 1)) {
           resultado = "Aprobado";
        } else if ((result == 2)) {
           resultado = "No aprobado";
        }
        Inspeccion aInspection = new Inspeccion(inspector, actividad, dia, mes,
horaReal, comentario, resultado, riesgoo, month);
        inspecciones.add(aInspection);

        System.out.println(aInspection + "\nLa inspeccion fue registrada con
exito!\n_____
_____");
        System.out.println("Desea registrar otra inspeccion ?\n1-Si\n2-No");
```

```
            No = Option1or2();
            in.nextLine();

        } while (No != 2);

    }

    private static void BajaDeInspector(ArrayList<Inspector> inspectores,
ArrayList<Inspeccion> inspecciones) {
        int opcion = 0;
        int numerador = 1;
        Inspector inspector = new Inspector();
        boolean ok = false;
        System.out.println("_____DAR DE BAJA A
INSPECTOR_____\n\nQue inspector desea dar de baja ? ");
        for (int i = 0; i < inspectores.size(); i++) {
            System.out.println(numerador + ")- " + inspectores.get(i).getNombre());

            numerador++;

        }
        opcion = Numbers();
        in.nextLine();
        while (!ok) {
            if ((opcion < 1) || (opcion >= numerador)) {
                System.out.println("Elija un inspector dentro del rango (1-" + (numerador - 1)
+ ")");
                opcion = Numbers();
                in.nextLine();
            } else {
```

```
            ok = true;
          }
        }
        boolean inspeccionHecha = false;
        numerador = opcion;
        inspector = inspectores.get((opcion - 1));
        for (int i = 0; i < inspecciones.size(); i++) {
          if (inspecciones.get(i).getInspector().equals(inspector)) {
            inspeccionHecha = true;


          }


        }
        if (inspeccionHecha) {
          System.out.println("El inspector " + inspector.getNombre() + " no puede ser
dado de baja porque tiene al menos una inspeccion realizada.");
        } else if (inspeccionHecha == false) {
          inspectores.remove(inspector);
          System.out.println("El inspector " + inspector.getNombre() + " fue dado de baja
exitosamente .");


        }


    }
    public void Programa(){
      Sistema sistema = new Sistema();
      menu(sistema);
    }
```

```
    private static void consultaSec(ArrayList<Inspeccion> inspecciones) {
        int[] secciones = new int[11];
        int a = 0;
        int na = 0;
        int max = 0;
        for (int i = 1; i < secciones.length; i++) {
            for (int j = 0; j < inspecciones.size(); j++) {
                if (inspecciones.get(j).getActividad().getSeccion() == (i) &&
(inspecciones.get(j).getResultado().equalsIgnoreCase("aprobado"))) {
                    a++;

                } else if (inspecciones.get(j).getActividad().getSeccion() == (i) &&
(inspecciones.get(j).getResultado().equalsIgnoreCase("no aprobado"))) {
                    na++;
                }

            }
            secciones[i] = (a + na);

System.out.println("_____\nInspecciones
aprobadas en seccion " + i + " : " + a + "\nInspecciones no aprobadas en  seccion " + i +
" : " + na + "\n_____");
            a = 0;
            na = 0;
        }

        for (int i = 1; i < secciones.length; i++) {
            int seccion = i;
            int inspeccion = secciones[i];
            if (secciones[i] > max) {
                max = secciones[i];
```

```
        }
        if (inspeccion == max && max > 1) {

            System.out.println("\n\nLa seccion con la mayor cantidad de inspecciones
hechas fue la " + i + " con " + max + " inspecciones hechas.\n");

        } else if ((inspeccion == max) && (max == 1)) {

            System.out.println("\n\nLa seccion con la mayor cantidad de inspecciones
hechas fue la " + i + " con " + max + " inspeccion hecha.\n");
        }

    }

}
private static int validarOpcionMenu(){
    int opcion= 0;
    boolean ok = false;
    opcion = Numbers();
    while(!ok){
        if((opcion < 1 ) || (opcion>9)){
            System.out.println("Elija una opcion valida (1-9)");
            opcion = Numbers();
        }else{
            ok = true;
        }

    }
    return opcion;
```

```
    }


    public static void menu(Sistema sistema) {
        int option = 0;
        boolean ok = false;

        do {
            System.out.println("BIENVENIDO AL SISTEMA DE GESTION DE RIESGOS LABORALES\n\n_____________________MENU PRINCIPAL_____________________\n\n1-Registrar Inspector/a\n2-Registrar Encargado/a\n3-Registrar Actividad\n4-Registrar Inspeccion\n5-Mostrar Inspecciones\n6-Listado de inspecciones no aprobadas dado un mes\n7-Consultar por seccion\n8-Baja de inspector\n9-Terminar ");
            System.out.print("Ingrese opcion : ");
            option = validarOpcionMenu();

            if (option == 3 && sistema.getEncargados().isEmpty()) {
                System.out.println("No se puede ingresar a esta opcion , no hay ningun encargado/a registrado/a");
                option = 10;
            } else if (((option == 4) && (sistema.getInspectores().isEmpty())) || ((option == 4) && (sistema.getActividades().isEmpty()))) {
                System.out.println("No se puede ingresar a esta opcion por falta de datos , no hay ningun inspector/a registrado/a , o ninguna actividad registrada.");
                option = 10;
            }
            if (sistema.getInspectores().isEmpty() && option == 8) {
                System.out.println("No hay ningun inspector registrado para dar de baja");
```

```
            option = 10;
        }
        if(sistema.getInspecciones().isEmpty() && ((option ==5) ||(option ==6) ||
(option==7))){
            System.out.println("No hay ninguna inspeccion registrada , no se puede
ejecutar la opcion seleccionada .");
            option=10;
        }

        switch (option) {

            case 1:
                in.nextLine();
                RegisterInspector(sistema.getInspectores() , sistema.getEncargados());
                backToMenu();

                break;
            case 2:
                in.nextLine();
                RegistrarEncargado(sistema.getEncargados(), sistema.getInspectores());
                backToMenu();

                break;

            case 3:

                in.nextLine();
                RegisterActivity(sistema.getEncargados(), sistema.getActividades());
                backToMenu();
```

```
            break;
        case 4:
            in.nextLine();
            //Inspectores,actividades , inspecciones
            RegisterInspection(sistema.getInspectores(), sistema.getActividades(),
sistema.getInspecciones());
            backToMenu();
            break;
        case 5:
            in.nextLine();
            //Inspecciones
            ShowInspections(sistema.getInspecciones());
            backToMenu();
            break;
        case 6:
            in.nextLine();
            //Inspecciones no aprobadas
             NonApprovedInspections(sistema.getInspecciones());
            in.nextLine();

            backToMenu();

            break;
        case 7:
            in.nextLine();
            consultaSec(sistema.getInspecciones());
            backToMenu();
            break;
        case 8:
            in.nextLine();
```

```
                //Inspector ,inspecciones
                BajaDeInspector(sistema.getInspectores(), sistema.getInspecciones());
                backToMenu();
                break;

            case 10: {
                in.nextLine();
                backToMenu();
                break;
            }

        }

    } while (option != 9);
  }

}
```

Diagrama UML: