SEED

"Simuladores para Estudio de Estructuras de Datos"

Manual de Usuario

Simulador Grafos<T>

Versión: 1.0

Universidad Francisco de Paula Santander
Programa Ingeniería de Sistemas
2014

MANUAL DE USUARIO: Simulador “GrafoND<T>”

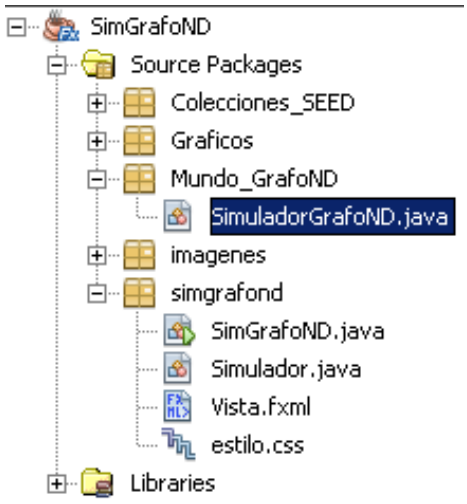
Descripción General

El presente Manual de usuario pretende describir en detalle el conjunto de funcionalidades de la Aplicación desarrollada para la Simulación del comportamiento de la Estructura de Datos GrafoND<T>.

Dentro de esta aplicación encontrará el estudiante un conjunto de operaciones relacionadas con las funciones básicas implementadas para la estructura Grafo No Dirigido y que pueden ser aplicables al Digrado: InsertarVértices, Insertar Aristas, EliminarVértices, Eliminar Aristas, conocer matriz de Adyacencia e Incidencia y Ruta entre un par de Nodos.

Adicionalmente el estudiante podrá reconocer la formación de rutas dentro del grafo, por medio de animaciones entre sus vértices como: Costo mínimo Dijkstra, Ciclo Euleriano, Ciclo Hamiltoniano, Recorrido en profundidad y en Anchura.

Para la implementación de este Simulador se ha determinado la siguiente distribución de paquetes, ya conocida por el Estudiante, de forma que sea fácilmente apropiable a futuras modificaciones con el fin de hacer buen uso de esta aplicación.



“Directorio del Simulador para GrafoND<T>”

La implementación de este simulador se realiza de una forma muy particular diferente a las demás. Los datos dentro del Grafo son insertados directamente con hacer un **clic** dentro del tablero, de forma que se puedan ir pintando a conveniencia cada vértice y arista determinada. La eliminación se realiza de

forma simular, indicando con un solo **clic** el objeto Vértice o Arista que desea eliminar, como se explicará más en detalle más adelante.

A continuación se presenta la interface principal del simulador para “GrafoND”.

“Interface principal del Simulador para GrafoND<T>”

Descripción de las Funcionalidades del Simulador

El simulador para GrafoND<T> permite al Estudiante:

1. **Insertar Datos:**
   1.1. Insertar Vértices
   Para insertar nuevo elementos vértices el estudiante debe primero oprimir el botón **insertar Vértice**, y dar clic en la **posición** del Área de dibujo donde desea que sea pintado el nuevo Vértice dentro del Grafo.
   .

"Inserción de 7 elementos Vértice dentro del Grafo"

1.2. Insertar Aristas

Para insertar nuevo elementos Aristas el estudiante debe primero oprimir el botón **insertar Arista**, y dar clic en la **posición** del primer vértice de la Arista (vértice inicial), luego debe seleccionar el segundo Vértice que hace parte de la Arista, en ese instante el grafo será pintado. (Previamente debe ser ingresado el peso de la Arista).

"Inserción de algunas Aristas para conectar el grafo creado."

2. **Eliminar Datos:**

   Para eliminar datos el estudiante deberá seleccionar el Botón**eliminar Vértice** o **eliminar Arista,** posteriormente se deberá dar clic en el elemento que desea eliminar, de esta forma el Grafo será actualizado con los datos restantes.

"Eliminación del Vértice D. El Grafo se actualiza inmediatamente"

3. **Representación del Grafo**

   Las dos opciones implementadas para la representación del Grafo son: La matriz de Adyacencia y la matriz de Incidencia.

   Para conocer la matriz de Adyacencia, el estudiante deberá dar **clic** en el botón **Matriz Adyacencia**, la cual reemplazará el Grafo en pantalla.

"Representación del grafo en una Matriz de Adyacencia"

Igualmente, para conocer la matriz de Incidencia, el estudiante deberá dar **clic** en el botón **Matriz Incidencia**, la cual reemplazará el Grafo en pantalla.

"Representación del grafo en una Matriz de Incidencia"

4. **Calcular las Ruta entre 2 Vértices**

   Para obtener la Ruta entre 2 Vértices, el estudiante debe diligenciar el campo con dos valores **v1-v2**, especificando los Vértices (separados por un guión). Ejemplo: “A-G”. Inmediatamente por medio de una animación la ruta entre los Vértices será pintada en el Grafo.

“Calcular la ruta entre los Vértices A-F”

5. **Calcular la Ruta menos Costosa entre Nodos “Dijkstra”**

   Para obtener la RutaDijkstra entre 2 Vértices, el estudiante debe diligenciar el campo con dos valores **v1-v2**, especificando los Vértices (separados por un guión). Ejemplo: “A-G”. Inmediatamente por medio de una animación la ruta entre los Vértices será pintada en el Grafo.

"Calcular la rutaDijkstra entre los Vértices A-F"

6. Determinar Ciclo y Camino Euleriano del Grafo

Para determinar el ciclo o camino Euleriano de un Grafo, el estudiante simplemente deberá dar clic al botón **ciclo Euleriano** o **Camino Euleriano**, dependiendo de la existencia de dicha consulta, el grafo mostrará por medio de una animación el ciclo o camino Euleriano del Grafo.

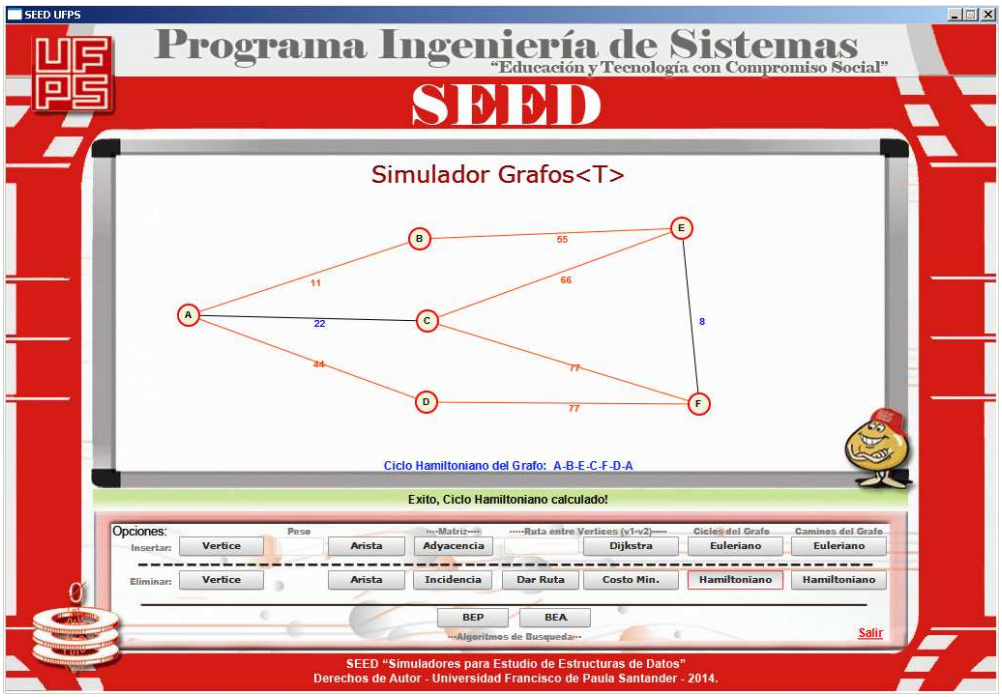
"Determinar ciclo Euleriano de un Grafo"

Ahora para determinar el camino Euleriano del mismo Grafo:

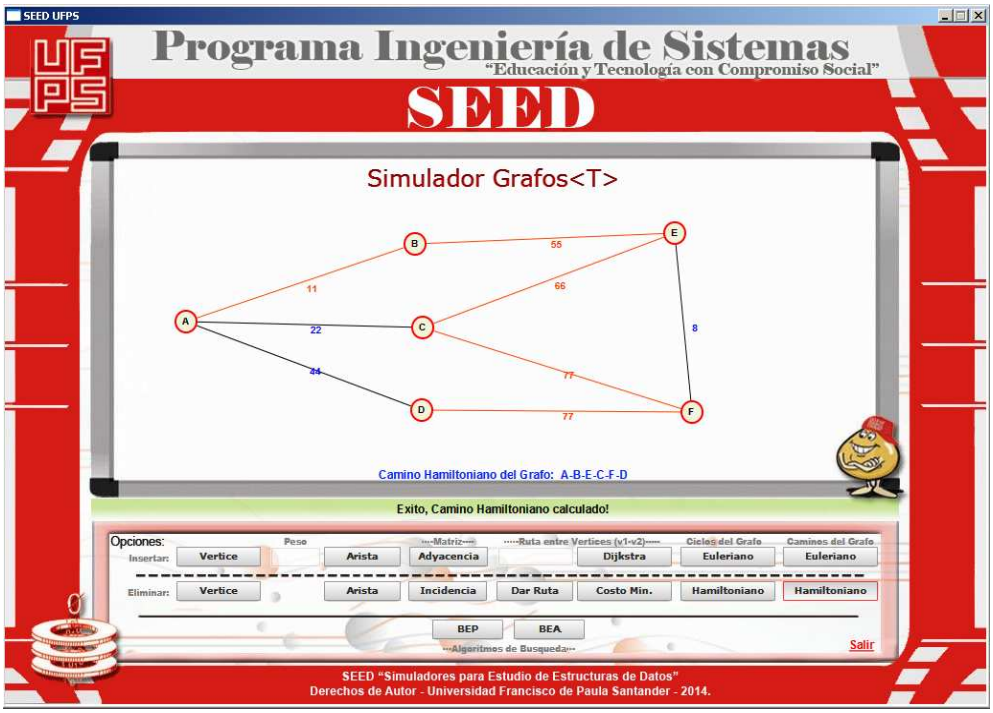
"Determinar camino Euleriano de un Grafo"

7. Determinar Ciclo y Camino Hamiltoniano del Grafo

Para determinar el ciclo o camino Hamiltoniano de un Grafo, el estudiante simplemente deberá dar clic al botón **ciclo Hamiltoniano** o **Camino Hamiltoniano**, dependiendo de la existencia de dicha consulta, el grafo mostrará por medio de una animación el ciclo o camino Hamiltoniano del Grafo.



Ahora para determinar el camino Hamiltoniano del mismo Grafo:

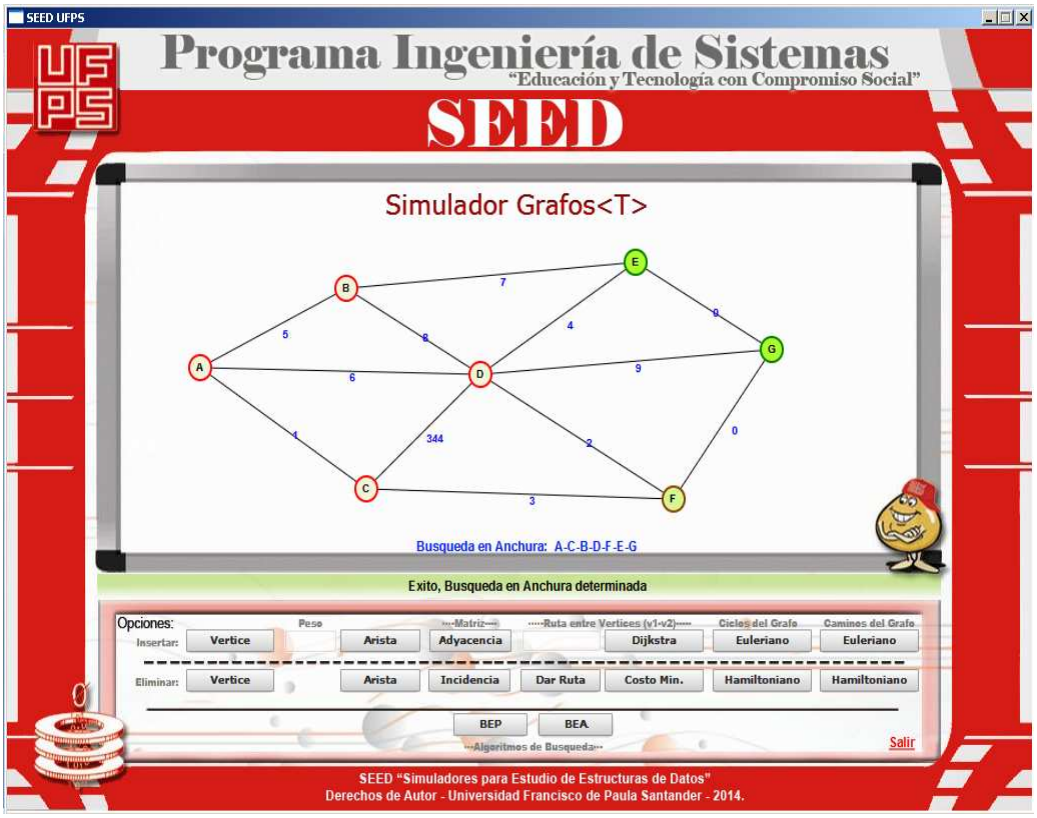


8. Recorridos en el Grafo

Para recorrer el Grafoel estudiante deberá seleccionar el recorrido que desea realizar para la búsqueda de los datos. La búsqueda se realizará a partir siempre del primer Vértice insertado y puede ser de 2 tipos: Búsqueda en Profundidad (BEP) o Búsqueda en Anchura (BEA).

Búsqueda en Profundidad

Búsqueda en Anchura



9. Adicionar nuevas funcionalidades:

Adicionalmente a las funciones incorporadas para el Simulador de GrafoND, existe la posibilidad de que el estudiante pueda **"adicionar nuevas funcionalidades"** a la aplicación, de acuerdo a las actividades asignadas por los docentes o el interés propio de generar nuevos algoritmos en cada estructura y poder simularlos gracias a la herramienta gráfica del Simulador.

A continuación se presentan los pasos que deberá seguir el estudiante para crear una nueva funcionalidad dentro del Simulador de GrafoND:

9.1. El estudiante debe generar el nuevo Algoritmo dentro de la Estructura de Datos**GrafoND**, presente en el paquete **SEED_Colecciones** y que desea adicionar a la funcionalidad del Simulador. (Para el ejemplo, se creará un algoritmo que permita determinar el listado de vecinos de un Vértice):

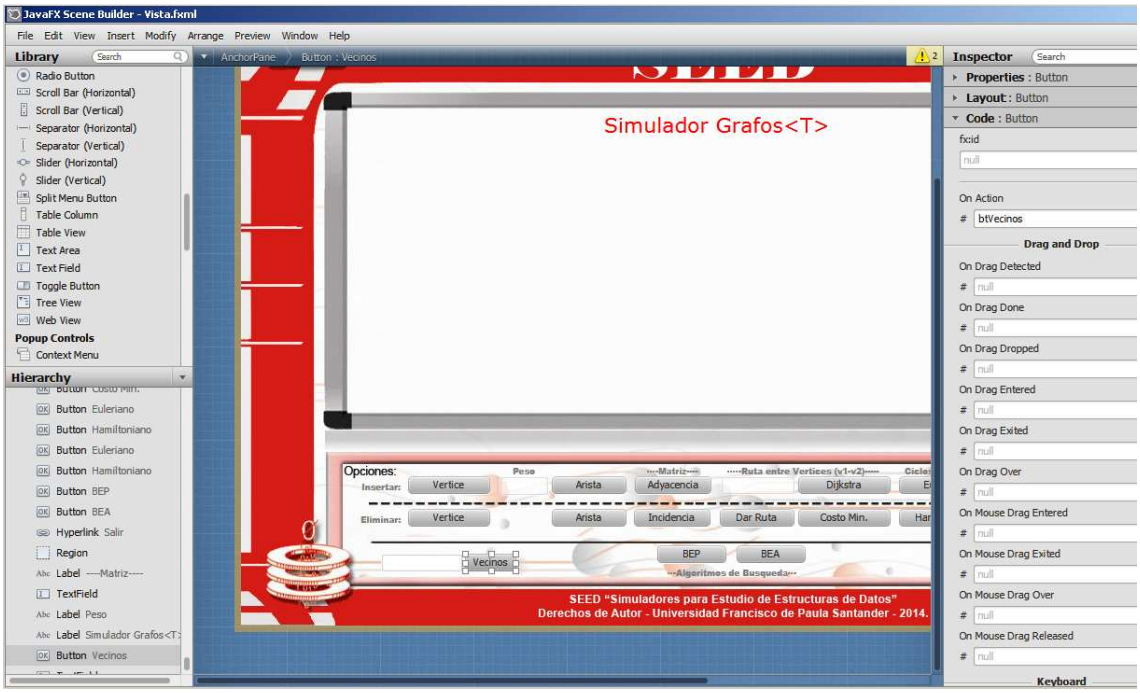
```java
/**
 * Metodo que permite conocer los vertices adyacentes/vecinos a un vertice indicado. <br>
 * <b>post:</b> Se retorno el conjunto de Vertices vecinos al info indicado. <br>
 * @param info Representa la informacion del Vertice que se desea evaluar. <br>
 * @return Un objeto de tipo ListaCD con los vertices vecinos al Vertice indicado.
 */
public ListaCD<Vertice> getVecinosVertice(T info){
    Vertice v = this.buscarVertice(info);
    if(v==null)
        return (null);
    return (v.getVecinos());
}
```

9.2. A continuación el Estudiante debe generar un Método en la clase **SimuladorGrafoND** del paquete **Mundo_GrafoND**, que realice el **llamado** al Método con el nuevo algoritmo creado en la Estructura de Datos **GrafoND**. Para el llamado debe utilizar el objeto creado en el Mundo **miGrafo**.

```java
public String listarVecinos(char v){
    String cad="";
    ListaCD<Vertice> l = this.miGrafo.getVecinosVertice(v);
    for(Vertice obj: l){
        cad+=obj.toString()+"-";
    }
    return (cad);
}
```

9.3. Por último, se deberá crear el componente grafico (para el ejemplo **Button**) que permita realizar el llamado al Método creado en **SimuladorGrafoND.** Existe dos posibilidades para ello: Utilizar la herramienta **"JavaFX SceneBuilder"** para insertarlo, o agregar el código del Button en el Archivo **Vista.fxml**.

9.3.1. Utilizando JavaFX SceneBuilder



9.3.2. Insertando directamente el elemento en Vista.fxml

```
<Button layoutX="240.0" layoutY="613.0" mnemonicParsing="false" onAction="#btVecinos" text="Vecinos" />
<TextField fx:id="txtVecinos" layoutX="150.0" layoutY="613.0" prefWidth="88.0" />
```

Es importante resaltar, para ambos casos, que se debe asignar el evento **“OnAction”** del Botón, para el ejemplo **“btVecinos”**, el cual será el nombre del **Método** dentro de la clase **Controlador** que permite realizar la nueva funcionalidad del Simulador. Igualmente la caja de texto **txtVecinos**deberá ser tenida en cuenta en la implementación.

A continuación el Método dentro del paquete **simgrafond**en la clase **Simulador** que permite realizar el llamado a la nueva funcionalidad.

```
@FXML    private      TextField    txtVecinos;


@FXML
private void btVecinos(){
    char val = this.txtVecinos.getText().charAt(0);
    String vec = this.simulador.listarVecinos(val);
    this.msg.setText(vec);
    this.msg.setVisible(true);
    this.impNota("Vecinos del Vertice determinados!", 0);
}
```

Además del llamado al Método creado en **SimuladorGrafoND**, el estudiante deberá invocar el método que le permita volver pintar el Árbol, el cual siempre será **"pintarTDA()"**. Opcionalmente se recomienda enviar una mensaje con la respuesta a la operación realizada utilizando **"impNota(" Mensaje a enviar " , tipo)"** donde tipo es cero **(0)** si en un mensaje Exitoso y uno **(1)** en caso de ser un mensaje erróneo.

A continuación se comprueba el funcionamiento del Algoritmo realizado:

"Se imprimió los vecinos de A del Grafo".