TALLER 1 JAVA DEL CURSO
INTEGRADOR DE SISTEMAS-SOFTWARE

FACULTAD : INGENIERÍA DE SISTEMAS Y ELECTRÓNICA
DIRECCIÓN : INGENIERÍA DE SISTEMAS
CURSO : INTEGRADOR SISTEMAS-SOFTWARE
DOCENTE : VARIOS

EXPERIENCIA DE PRÁCTICAS
DURACIÓN: 4 HORAS ACADEMICAS

TEMA: **VARIOS**

OBJETIVOS:

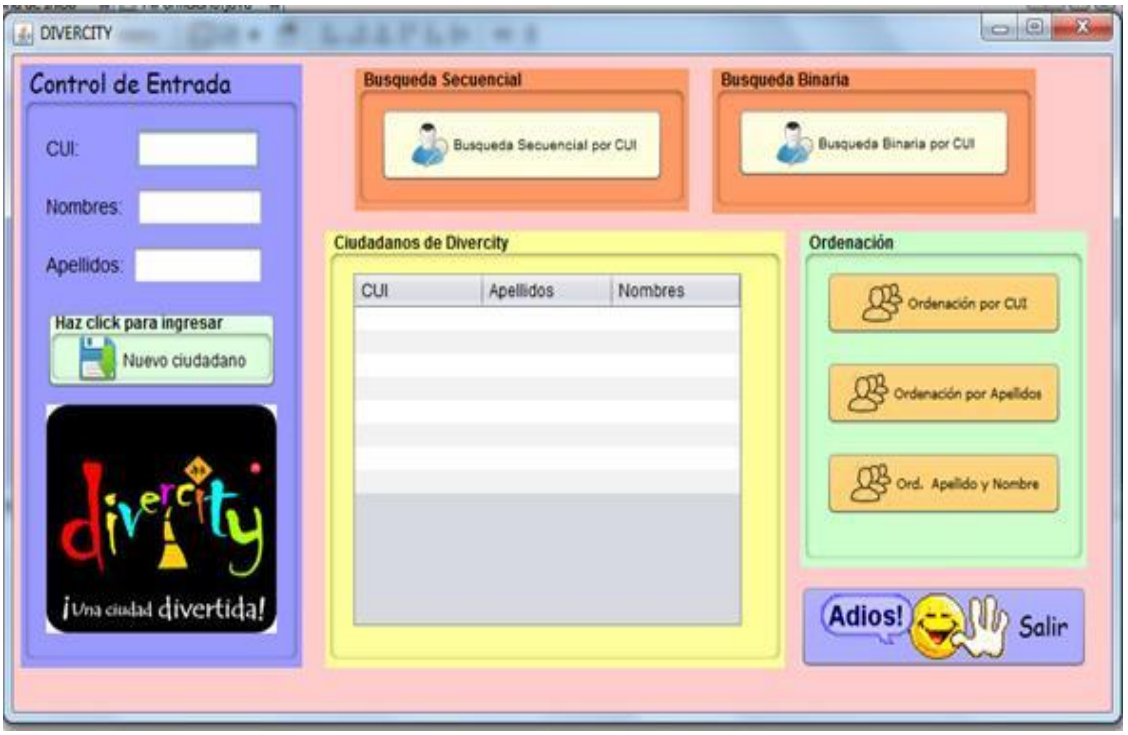
- Repasar los conceptos necesarios para desarrollar aplicaciones utilizando programación orientada a objetos y el entorno Java,

SOFTWARE A UTILIZAR: NETBEANS.

Tema : Métodos de Ordenamiento y Búsqueda

1. CASO DE ESTUDIO: PROYECTO GESTION DIVERCITY

Desarrollar una aplicación Java Swing que permita gestionar el ingreso de niños (ciudadanos) a DiverCity. La interface visual deberá tener la siguiente funcionalidad:



La aplicación debe tener las siguientes funcionalidades:

- Controlar el ingreso de ciudadanos a DiverCity,
- Conforme un ciudadano ingresa se debe mostrar los datos en una lista (JTable).
- Ordenar ascendentemente por código único de identidad (cui) a los ciudadanos.
- Ordenar por apellidos a todos los ciudadanos.
- Realizar una doble ordenación, si hay ciudadanos con los mismos apellidos ordenarlos a estos por nombres.
- Ubicar a un ciudadano por cui. Utilizar búsqueda secuencial.
- Ubicar a un ciudadano por cui .Utilizar búsqueda binaria.

Para los casos de ordenamiento, utilizar el método refinado de la Burbuja.

CONSIDERE PARA SU SOLUCIÓN:

- Para la solución del caso de negocio se deberá diseñar clases cuyos métodos permitan realizar un ordenamiento por cui, por apellidos y una doble ordenación por apellidos y nombres, utilizando el método mejorado de la Burbuja. Los métodos también deberán permitir ubicar al ciudadano con un cui determinado, realizando una búsqueda secuencial y una búsqueda binaria en DiverCity.

- La clase **Ciudadano** tendrá como variables de instancia (atributos) el código único de identidad (cui), nombres y apellidos. El método constructor con parámetro y los métodos getters y setters.

- La clase **DiverCity** tendrá como variable de instancia un ArrayList de tipo Ciudadano

- La aplicación deberá permitir crear un arreglo de referencias con "referencias" a objetos tipo Ciudadano.

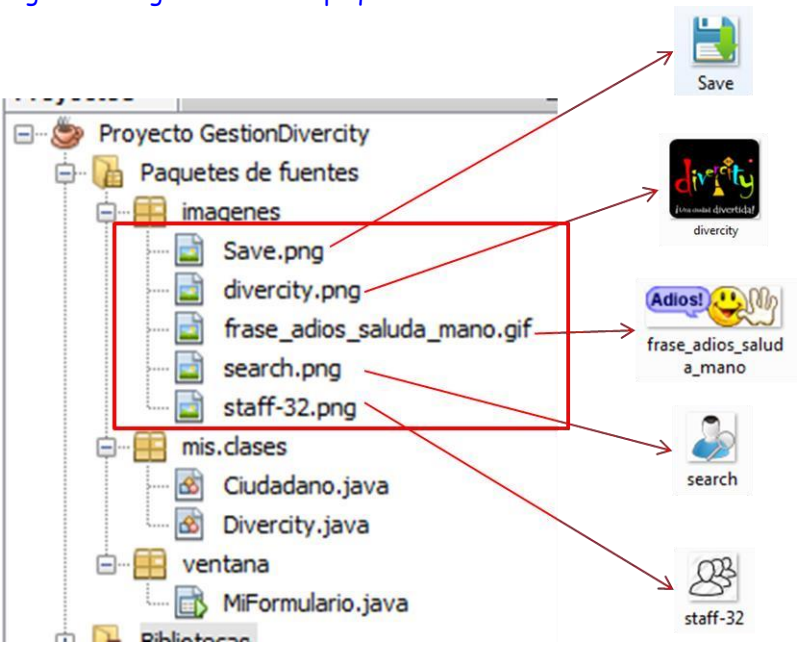
- El arreglo de referencias que "referencian" a los objetos ciudadanos de DiverCity, se deberá ordenar ascendentemente según las variables de instancias indicadas, utilizando el método mejorado de la Burbuja.

- En el arreglo de referencias se deberá realizar la búsqueda secuencial y la búsqueda binaria para ubicar la referencia del objeto Ciudadano buscado.

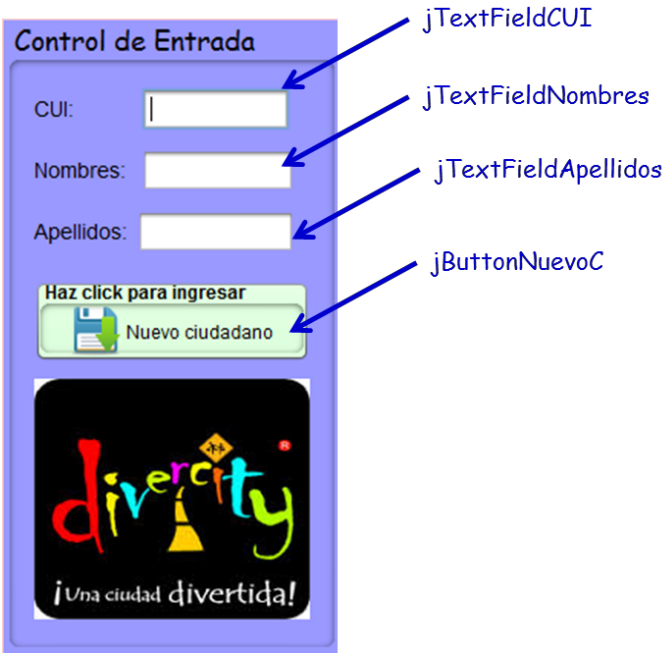
- Toda la información obtenida se deberá mostrar en el componente JTable.

Solución:

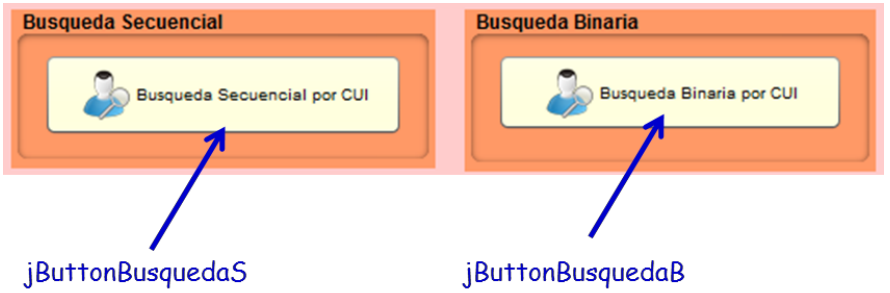
1. Considere la siguiente organización de paquetes:



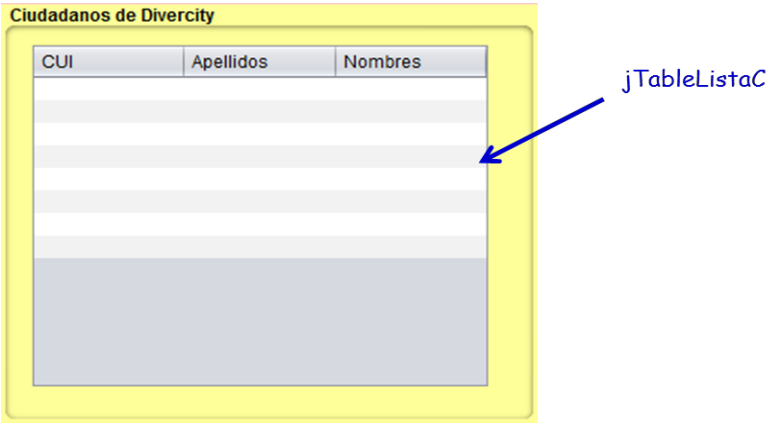
2. Diseño del Panel Control de Entrada:



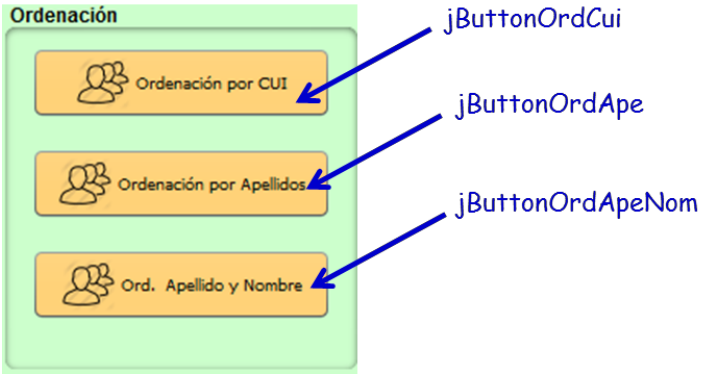
3. Diseño de los paneles Búsqueda Secuencial y Búsqueda Binaria:



4. Diseño del panel Ciudadanos de Divercity:



5. Diseño del Panel Ordenación:



6. Diseño del Botón Salir:

7. Diseño de la clase Ciudadano:

```
1   package mis.clases;
2
3   public class Ciudadano {
4       private int cui;
5       private String nombres;
6       private String apellidos;
7
8       public Ciudadano(int cui, String nombres, String apellidos) {
9           this.cui = cui;
10          this.nombres = nombres;
11          this.apellidos = apellidos;
12      }
13
14      public int getCui() {...3 lines }
17
18      public void setCui(int cui) {...3 lines }
21
22      public String getNombres() {...3 lines }
25
26      public void setNombres(String nombres) {...3 lines }
29
30      public String getApellidos() {...3 lines }
33
34      public void setApellidos(String apellidos) {
35          this.apellidos = apellidos;
36      }
37  }
```

8. Diseño de la clase Divercity:

```
1   package mis.clases;
2
3   import java.util.ArrayList;
4   import javax.swing.JOptionPane;
5
6   public class Divercity {
7       private ArrayList<Ciudadano> listaC;
8
9       public Divercity() {
10          listaC=new ArrayList<>();
11      }
12
13      public ArrayList<Ciudadano> getListaC() {
14          return listaC;
15      }
16
17      public void ingresarCiudadano(Ciudadano refC){
18          Ciudadano objC=busquedaS(refC.getCui());
19
20          if(objC==null){
21              listaC.add(refC);
22              JOptionPane.showMessageDialog(null, "El ciudadano ha ingresado ");
23          }else{
24              JOptionPane.showMessageDialog(null, "Ciudadano ya esta en Divercity");
25          }
26      }
27
28      public Ciudadano busquedaS(int cod){
29          Ciudadano refC=null;
30
31          boolean estado =false;
32          int i=0;
33
34          while(i<listaC.size() && estado==false){
35              if(cod==listaC.get(i).getCui()){
36                  estado=true;
37                  refC=listaC.get(i);
38              }else{
39                  i++;
40              }
41          }
42          return refC;
43      }
```

```
45      public void ordenarCiudadanosAscPorCui (){
46          int i,j;
47          Ciudadano aux;
48
49          for(i=0;i<listaC.size()-1;i++){
50              for(j=i+1;j<listaC.size();j++){
51                  if(listaC.get(i).getCui()>listaC.get(j).getCui()){
52                      aux=listaC.get(i);
53                      listaC.set(i, listaC.get(j));
54                      listaC.set(j, aux);
55                  }
56              }
57          }
58      }
59
60      public void ordenarCiudadanosPorApellidos (){
61          int i,j;
62          Ciudadano aux;
63
64          for(i=0;i<listaC.size()-1;i++){
65              for(j=i+1;j<listaC.size();j++){
66                  if(listaC.get(i).getApellidos().compareToIgnoreCase(listaC.get(j).getApellidos()) >0){
67                      aux=listaC.get(i);
68                      listaC.set(i, listaC.get(j));
69                      listaC.set(j, aux);
70                  }
71              }
72          }
73      }
75      public void ordenarCiudadanosPorApellidosYNombres (){
76          int i,j;
77          Ciudadano aux;
78
79          for(i=0;i<listaC.size()-1;i++){
80              for(j=i+1;j<listaC.size();j++){
81                  if(listaC.get(i).getApellidos().compareToIgnoreCase(listaC.get(j).getApellidos()) >=0
82                          && listaC.get(i).getNombres().compareToIgnoreCase(listaC.get(j).getNombres())>0){
83                      aux=listaC.get(i);
84                      listaC.set(i, listaC.get(j));
85                      listaC.set(j, aux);
86                  }
87              }
88          }
89      }
90      public Ciudadano busquedaBinariaPorCui(int cuiBusq){
91          Ciudadano refCiuBusq=null;
92          int cen=0,izq=0 ,der=listaC.size()-1;
93
94          while (izq<=der && refCiuBusq==null){
95              cen = (izq+der)/2;
96              if(cuiBusq==listaC.get(cen).getCui()){
97                  refCiuBusq = listaC.get(cen);
98              } else if (cuiBusq< listaC.get(cen).getCui()){
99                  der=cen-1;
100             } else{
101                 izq=cen+1;
102             }
103         }
104         return refCiuBusq;
105     }
```

9. Diseño de la clase MiFormulario:

```
1   ...4 lines
5   package ventana;
6
7   import javax.swing.JOptionPane;
8   import mis.clases.Ciudadano;
9   import mis.clases.Divercity;
10
11  public class MiFormulario extends javax.swing.JFrame {
12      Divercity refG;
13
14      /** Creates new form MiFormulario ...3 lines */
17
18      public MiFormulario() {
19          initComponents();
20          refG=new Divercity();
21      }
22
23      /** This method is called from within the constructor to initialize the form ...5 lines */
28      @SuppressWarnings("unchecked")
29      Generated Code
346
347     //Métodos de apoyos
348     private void listarCiudadanos(){
349         for(int i=0; i<refG.getListaC().size();i++){
350                 jTableListaC.setValueAt(refG.getListaC().get(i).getCui(), i, 0);
351                 jTableListaC.setValueAt(refG.getListaC().get(i).getApellidos(), i, 1);
352                 jTableListaC.setValueAt(refG.getListaC().get(i).getNombres(), i, 2);
353         }
354     }
356     private void mostrarCiudadanoBuscadoPorCui(Ciudadano refC,int cuiBusq){
357       if(refC !=null){
358             JOptionPane.showMessageDialog(null,"El ciudadano con CUI   " +
359              refC.getCui() + " es  " +refC.getNombres() + "  " +refC.getApellidos());
360       } else  {
361           JOptionPane.showMessageDialog(null,"El ciudadano con CUI  " +
362                           cuiBusq + "   no ingreso a DiverCity");
363       }
364     }
365
366     private void jButtonBusquedaSActionPerformed(java.awt.event.ActionEvent evt) {
367         int cuiBusq;
368         //buscar secuencialmente a un ciudadano
369         //con un cuiBusq determinado
370         cuiBusq=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingrese un cui a buscar : "));
371         //me devuelve la referencia del niÃ±o buscado
372         Ciudadano refC=refG.busquedaS(cuiBusq);
373         mostrarCiudadanoBuscadoPorCui(refC, cuiBusq);
374     }
```

```
376     private void jButtonNuevoCActionPerformed(java.awt.event.ActionEvent evt) {
377         int ciu;
378         String nombre,ape;
379         Ciudadano refCiu;
380 
381         ciu=Integer.parseInt(jTextFieldCUI.getText());
382         nombre=jTextFieldNombres.getText();
383         ape=jTextFieldApellidos.getText();
384 
385         refCiu=new Ciudadano(ciu, nombre, ape);
386         refG.ingresarCiudadano(refCiu);
387         listarCiudadanos();
388 
389         //borrar datos de los jTextFields
390         jTextFieldNombres.setText("");
391         jTextFieldApellidos.setText("");
392         jTextFieldCUI.setText("");
393     }
394 
395     private void jButtonSalirActionPerformed(java.awt.event.ActionEvent evt) {
396         this.dispose();
397     }
399     private void jButtonBusquedaBActionPerformed(java.awt.event.ActionEvent evt) {
400         int cuiBusq;
401         cuiBusq=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingrese un cui a buscar : "));
402 
403         //el arreglo de referencias tiene que estar ordenado por cui
404         refG.ordenarCiudadanosAscPorCui();
405         Ciudadano refC=refG.busquedaBinariaPorCui(cuiBusq);
406         mostrarCiudadanoBuscadoPorCui(refC,cuiBusq);
407     }
408 
409     private void jButtonOrdCuiActionPerformed(java.awt.event.ActionEvent evt) {
410 
411         refG.ordenarCiudadanosAscPorCui();
412         listarCiudadanos();
413     }
414 
415     private void jButtonOrdApeActionPerformed(java.awt.event.ActionEvent evt) {
416 
417         refG.ordenarCiudadanosPorApellidos();
418         listarCiudadanos();
419     }
420 
421     private void jButtonOrdApeNomActionPerformed(java.awt.event.ActionEvent evt) {
422 
423         refG.ordenarCiudadanosPorApellidosYNombres();
424         listarCiudadanos();
425     }
```