

# Documento soporte

*Juan Esteban Carrillo Garcia, Hamilton Espitia Rozo, Maria Fernanda Perez Hernandez*

## **Diseño de la solución:**

Se planteó una solución al problema utilizando estructuras como las listas enlazadas y las colas dobles. Estas estructuras fueron desarrolladas a lo largo del curso y se utilizaron en conjunto para facilitar el manejo de la información en el programa. A su vez para la implementación de las consultas se usaron árboles RojiNegros y un árbol de consultas creado a partir de los conceptos de un árbol n-ario. Además, se hizo uso de archivos planos para facilitar la lectura y escritura de la información.

Iniciamos con la creación de las listas de las ciudades, partidos y candidatos. Se hace uso de punteros para almacenar y manipular las listas de candidatos, ciudades y partidos respectivamente. Los punteros a listas permiten la manipulación de las mismas, mientras que los punteros a objetos se utilizan para crear los objetos correspondientes y luego insertarlos en las listas.

Hacemos uso de una clase "Archivos" que a partir de una serie de funciones para la manipulación de los archivos relacionados con la registraduría, con la lectura, escritura, modificación y eliminación de contenido.

También tenemos una clase simulación que es llamada en el main, esta contiene funciones para mostrar los diferentes menús y realizar operaciones relacionadas con la simulación de las elecciones, como leer archivos, realizar consultas, mostrar estadísticas, realizar simulaciones y realizar inserciones, modificaciones y eliminaciones de datos. El objeto Inicializar se utiliza para inicializar los objetos de ciudades, partidos y candidatos.

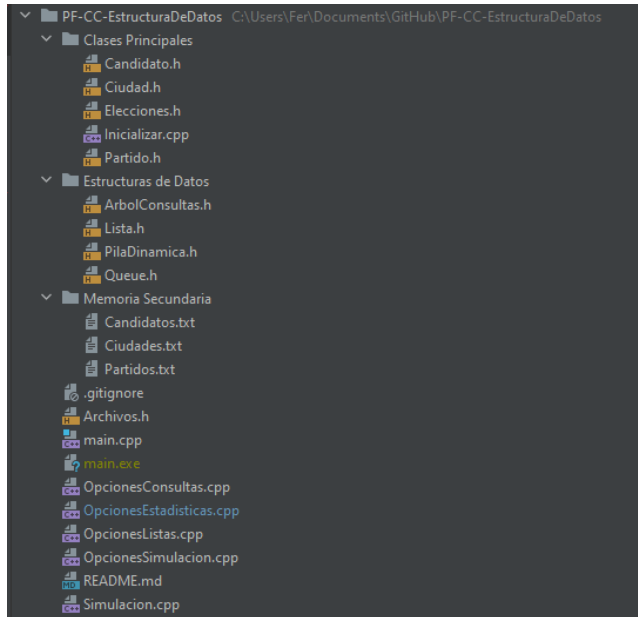
Para las listas tenemos una clase Opciones Listas con funciones que se encargan de mostrar información sobre ciudades, partidos y candidatos a partir de recibir punteros de las listas o los objetos.

Para las consultas se realizan sobre una estructura de datos llamada ArbolConsultas y muestra los resultados en la consola. Las consultas están diseñadas para obtener información sobre los candidatos, partidos y ciudades relacionados con las elecciones. Se tiene a listaArboles que es un puntero a una lista de NodoCiudad, se inicializa y se llenan los árboles de consulta con información de las ciudades, partidos y candidatos pasados como argumentos. Luego tenemos unas funciones para cada una de las consultas propuestas.

Para las estadísticas hay unas funciones que se encargan de mostrar información sobre los votos totales, votos por partido y votos por género en diferentes contextos, como por ciudad, por tipo de elección o a nivel nacional. En estadísticaPorCiudad se muestra un informe de votos para cada elección relacionada con esa ciudad. En estadísticaEleccion muestra las estadísticas de votos por tipo de elección (como "Alcaldia" o "Consejo") para una ciudad en específico. En estadísticaNacional muestra las estadísticas de votos a nivel nacional para un tipo de elección específico. Dando un informe de votos para cada partido político y

el total de votos, tanto para hombres como para mujeres, y finalmente agregarVotosPartido se utiliza para mantener un registro de los votos por partido en la lista listaPartidoVotos.

## Contenido:



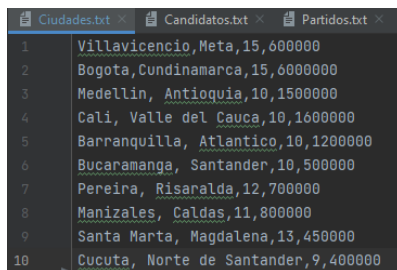
El programa utiliza archivos planos en la memoria secundaria para almacenar datos, mientras que las estructuras de datos como listas, estructuras, colas dobles, pilas y árboles se utilizan en la memoria principal para facilitar consultas, listas y estadísticas relacionadas con las elecciones simuladas. Además, el programa cuenta con un menú principal y varios submenús que permiten una navegación fluida por el sistema. El objetivo del programa es gestionar y analizar información electoral.

En general, el programa busca proporcionar una experiencia de usuario fluida y brindar información clara y organizada sobre los resultados de las elecciones simuladas.

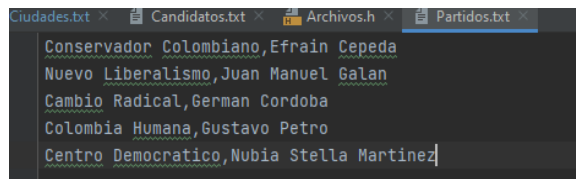
## Estructura de archivos planos:

Los archivos planos tienen diferentes estructuras dependiendo de su contenido y se encuentran de las siguientes maneras:

*Ciudad(nombre, departamento, tamConcejo, censoElectoral)*



*Partido(nombre, representanteLegal)*



```
Ciudades.txt x Candidatos.txt x Archivos.h x Partidos.txt x
Conservador Colombiano,Efrain Cepeda
Nuevo Liberalismo,Juan Manuel Galan
Cambio Radical,German Cordoba
Colombia Humana,Gustavo Petro
Centro Democratico,Nubia Stella Martinez
```

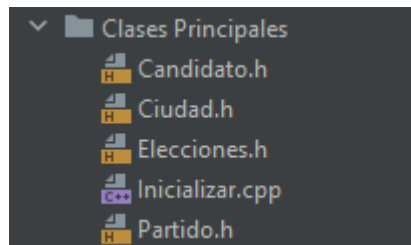
*Candidato(nombre, apellido, puesto, numIdentificacion, sexo, estadoCivil,fechaNacimiento, ciudadNacimiento, ciudadResidencia, partido, cantVotos)*



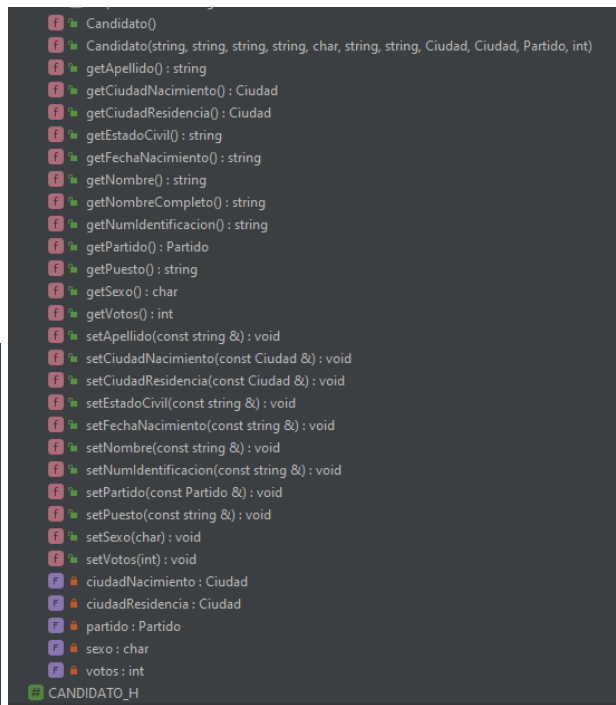
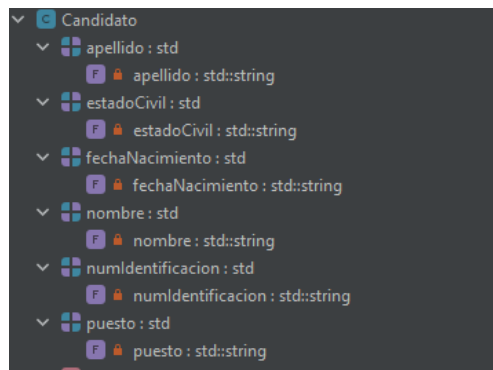
```
Ciudades.txt x Candidatos.txt x Archivos.h x Partidos.txt x
1 Juan,Garcia,Concejo,1003516421,M,Soltero,09/02/2002,Fusagasuga,Bogota,Nuevo Liberalismo,0
2 Jose,Aljure,Concejo,1205416421,M,Soltero,07/04/2003,Cali,Bogota,Centro Democratico,0
3 Lina,Carrillo,Alcaldia,2303516443,F,Soltera,05/07/2001,Medellin,Medellin,Nuevo Liberalismo,0
4 Sara,Cardona,Concejo,3563516421,F,Soltera,12/12/1986,Santa Marta,Fusagasuga,Nuevo Liberalismo,0
5 Pedro,Sanchez,Alcaldia,1144712521,M,Soltero,28/06/2003,Medellin,Cali,Conservador Colombiano,0
6 Rafael,Hernandez,Alcaldia,1756765344,M,Casado,08/11/2000,Bogota,Santa Marta,Colombia Humana,0
7 Jose Antonio,Cruz,Concejo,1251143921,M,Casado,27/07/1997,Santa Marta,Pereira,Colombia Humana,0
8 Jose Antonio,Lopez,Concejo,1414248750,M,Soltero,15/02/1970,Cali,Medellin,Nuevo Liberalismo,0
9 Jose,Gomez,Alcaldia,1616558227,M,Soltero,23/03/2003,Villavicencio,Barranquilla,Nuevo Liberalismo,0
10 Manuel,Lopez,Concejo,1278579226,M,Casado,05/03/1981,Villavicencio,Bogota,Conservador Colombiano,0
11 Jose Luis,Martinez,Concejo,1947031111,M,Soltero,18/05/1970,Santa Marta,Medellin,Colombia Humana,0
```

## Estructuras de datos en memoria principal definidas y diagrama:

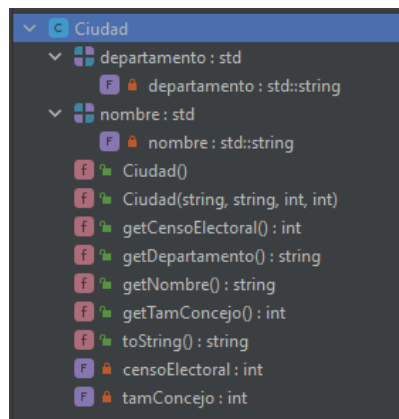
### Clases principales



Las funciones usadas al interior de Candidato fueron:



Las funciones usadas al interior de Ciudad fueron:

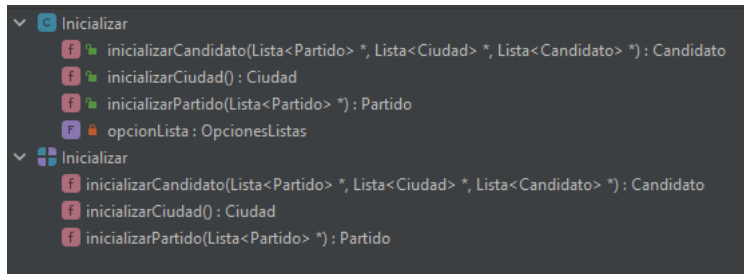


Las funciones usadas al interior de Partido fueron:

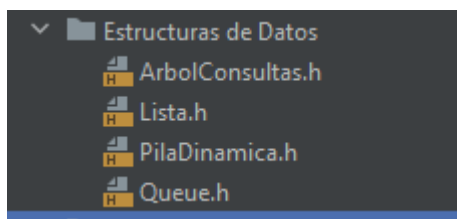


Las funciones usadas al interior de Elecciones fueron:

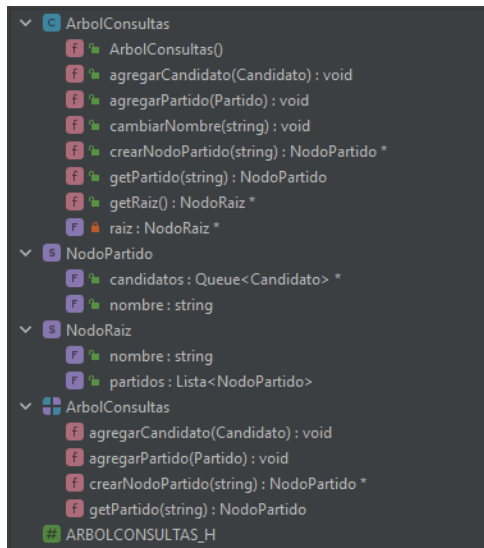
Las funciones usadas al interior de Inicializar fueron:



## Estructuras de datos



Las funciones usadas al interior de ArbolConsultas fueron:



Las funciones usadas al interior de Lista fueron:

```

v Lista<T>
  f Lista()
  f ~Lista() : void
  f borrar(int) : void
  f buscar(int) : T
  f existe(T) : bool
  f getTam() : int
  f insertar(T) : void
  f lista_vacia() : bool
  f modificar(T, int) : void
  F fin : Nodo<T> *
  F inicio : Nodo<T> *
  F tam : int
v C Nodo<T>
  f Nodo(T)
  F siguiente : Nodo<T> *
  F valor : T
## LISTA_H

```

Las funciones usadas al interior de PilaDinamica fueron:

```

v C Pila<T>
  > S nodoPila
  S nodoPila
  f Pila()
  f ~Pila() : void
  f estaVacia() : bool
  f meter(T) : void
  f sacar() : T
  f vacia() : int
  F cabeza : nodoPila *
  F z : nodoPila *
## PILADINAMICA_H

```

Las funciones usadas al interior de Queue fueron:

```

v S NodoQueue<T>
  F ant : NodoQueue<T> *
  F dato : T
  F sig : NodoQueue<T> *
v C Queue<T>
  f Queue()
  f ~Queue() : void
  f Dequeue(char) : void
  f Enqueue(T, char) : void
  f getTam() : int
  f queue_vacia() : bool
  f retornarElemento(int, char) : T
  F cab : NodoQueue<T> *
  F final : NodoQueue<T> *
  F tam : int
v Queue<T>
  f ~Queue() : void
  f Dequeue(char) : void
  f Enqueue(T, char) : void
  f queue_vacia() : bool
  f retornarElemento(int, char) : T
## queue_h

```

Las funciones usadas al interior de Archivo.h fueron:

```
Archivos
  Archivos(string)
  anadir(string) : void
  eliminar(string) : void
  escribir(string) : void
  leerCandidatos() : Lista<Candidato> *
  leerCiudades() : Lista<Ciudad> *
  leerPartidos() : Lista<Partido> *
  modificar(string, string) : void
  candidatos : Lista<Candidato> *
  ciudades : Lista<Ciudad> *
  nombreArchivo : string
  partidos : Lista<Partido> *
  rutaArchivo : string
  ARCHIVOS_H
```

Las funciones usadas al interior de OpcionesConsultas.cpp fueron:

```
NodoCiudad
  arbol : ArbolConsultas *
  nombre : string
OpcionesConsultas
  OpcionesConsultas(Lista<Ciudad> *, Lista<Partido> *, Lista<Candidato> *)
  calcularEdad(Candidato) : int
  consulta1(string, string) : void
  consulta4(string) : void
  consulta5(string) : void
  consulta6(string) : void
  consulta7(Lista<Ciudad>) : void
  getList() : Lista<NodoCiudad> *
  listaArboles : Lista<NodoCiudad> *
OpcionesConsultas
  consulta1(string, string) : void
  consulta4(string) : void
  consulta5(string) : void
  consulta6(string) : void
  consulta7(Lista<Ciudad>) : void
```

Las funciones usadas al interior de OpcionesEstadisticas.cpp fueron:

```
OpcionesEstadisticas
  OpcionesEstadisticas(Lista<Ciudad> *, Lista<Partido> *, Lista<Candidato> *, Lista<Elecciones> *)
  agregarVotosPartido(string, int) : void
  estadisticaEleccion(string) : void
  estadisticaNacional(string) : void
  estadisticaPorCiudad() : void
  porcentaje(double, double) : double
  candidatos : Lista<Candidato> *
  ciudades : Lista<Ciudad> *
  listaCandidatosFiltrada : Lista<Candidato>
  listaPartidoVotos : Lista<PartidoVotos>
  opcionLista : OpcionesListas
  partidos : Lista<Partido> *
  posiciones : Lista<int>
  totalElecciones : Lista<Elecciones> *
PartidoVotos
  partido : string
  votos : int
```

Las funciones usadas al interior de OpcionesListas.cpp fueron:

```
▼ OpcionesListas
  f  candidatosAlcaldia(string, Lista<Candidato> *) : void
  f  candidatosAlcaldiaConsejoPartido(string, string, Lista<Candidato> *) : void
  f  candidatosAlcaldiaConsejoPartidoLista(string, Lista<Candidato> *, Lista<Partido> *) : void
  f  candidatosConcejo(string, Lista<Candidato> *) : void
  f  mostrarCandidatos(Lista<Candidato> *) : void
  f  mostrarCiudades(Lista<Ciudad> *) : void
  f  mostrarPartidos(Lista<Partido> *) : void
```

Las funciones usadas al interior de OpcionesSimulaciones.cpp fueron:

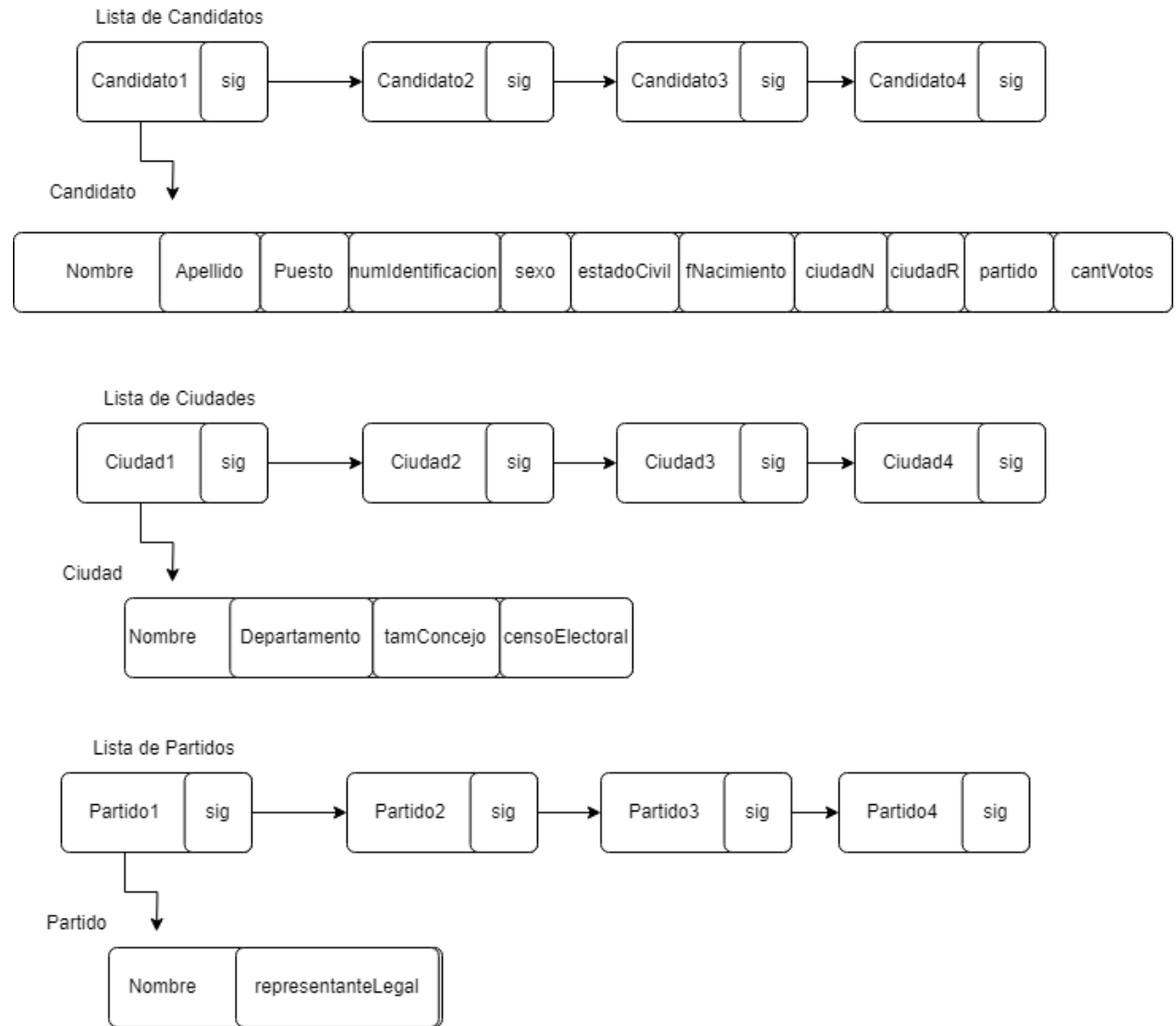
```
▼ OpcionesSimulacion
  f  OpcionesSimulacion(Lista<Ciudad> *, Lista<Partido> *, Lista<Candidato> *, Lista<Elecciones> *)
  f  asignacionVotos(int, int, int, int, int, Ciudad, string) : void
  f  candidatosPorCiudadPuesto(string, string) : void
  f  numeroAleatorio(int) : int
  f  porcentaje(double, double) : double
  f  simularVotos() : void
  F  candidatos : Lista<Candidato> *
  F  ciudades : Lista<Ciudad> *
  F  ganador : int[2]
  F  listaCandidatosFiltrada : Lista<Candidato>
  F  opcionLista : OpcionesListas
  F  partidos : Lista<Partido> *
  F  posiciones : Lista<int>
  F  totalElecciones : Lista<Elecciones> *
```

Las funciones usadas al interior de Simulaciones.cpp fueron:

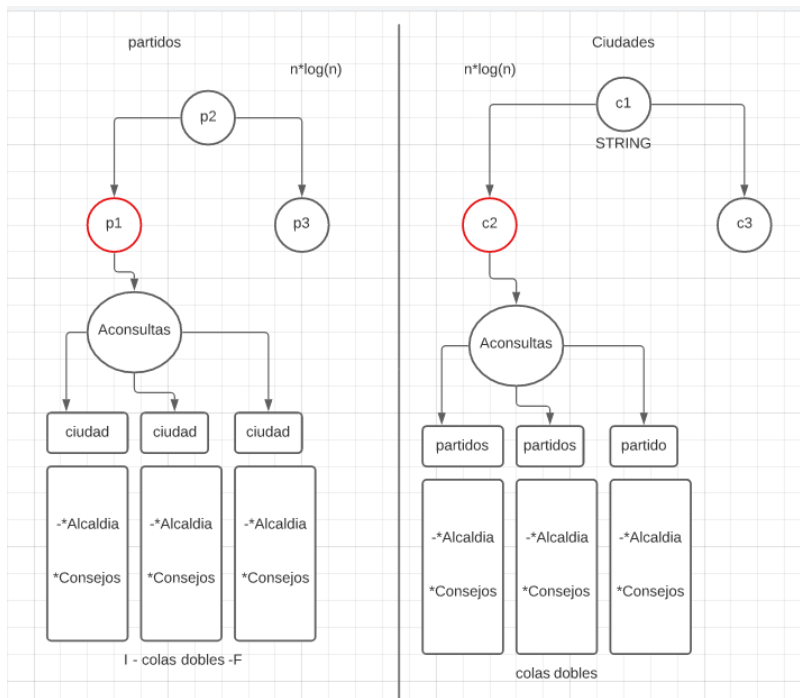
```
▼ Simulacion
  f  EleccionSubMenuInsercion() : void
  f  leerArchivos() : void
  f  Menu() : void
  f  MostrarMenu() : void
  f  SubMenuConsultas() : void
  f  SubMenuEstadisticas() : void
  f  SubMenuInsercion() : void
  f  SubMenuListas() : void
  F  candidatos : Lista<Candidato> *
  F  ciudades : Lista<Ciudad> *
  F  inicializar : Inicializar
  F  Opcion : int
  F  partidos : Lista<Partido> *
  F  totalElecciones : Lista<Elecciones> *
▼ Simulacion
  f  EleccionSubMenuInsercion() : void
  f  Menu() : void
  f  MostrarMenu() : void
  f  SubMenuConsultas() : void
  f  SubMenuEstadisticas() : void
  f  SubMenuInsercion() : void
  f  SubMenuListas() : void
```



## Listas de Candidatos, ciudades y partidos



## ArbolConsulta.



## Diagrama UML:

\* Archivo adjunto aparte

## Diagrama Casos de Uso:

