EjercicioListasAdyacencia

Estructuras de Datos

Tema 5: grafos

1º Grado en Ingeniería de la Computación
© Profesor Dr. Carlos Grima Izquierdo (www.carlosgrima.com)
URJC (www.urjc.es)

Programa en C++ la clase "EspacioAereo", representada por un grafo de aeropuertos interconectados mediante rutas aéreas. El grafo se implementará mediante listas de adyacencia. Es un grafo dirigido, no acíclico, no multigrafo, no pseudografo, no necesariamente conexo, etiquetado con los kilómetros de cada ruta aérea y la compañía que opera dicha ruta. Por cada aeropuerto, guardaremos una lista de sus rutas salientes y otra lista con sus rutas entrantes.

Cada nodo del grafo será un struct "Aeropuerto" que contendrá lo siguiente:

- El nombre del aeropuerto (puntero a un objeto de la clase string). Es la clave del aeropuerto.
- Un puntero a una lista enlazada de rutas salientes del aeropuerto.
- Un puntero a una lista enlazada de rutas entrantes del aeropuerto.

Cada arco del grafo será un struct "Ruta" que contendrá lo siguiente:

- El nombre del aeropuerto (un puntero a un objeto de la clase string). Este aeropuerto será el destino si este arco está en la lista enlazada de rutas salientes, o será el origen si este arco está en la lista enlazada de rutas entrantes. Esta información es la clave del arco.
- El número de kilómetros de la ruta.
- Un puntero a un objeto de la clase string con el nombre de la empresa que opera dicha ruta.

Los nodos del grafo (los aeropuertos) los guardaremos en una lista contigua. Para ello, reutilizaremos la lista contigua del EjercicioListaContigua2. Modificaremos la clase ListaContigua para que ahora guarde objetos de tipo Aeropuerto en vez de números enteros. También modificaremos el método "buscar()" para que busque únicamente por el nombre del aeropuerto.

Los arcos del grafo (las rutas aéreas) las guardaremos en las listas enlazadas. Para ello, vamos a reutilizar los elementos del EjercicioDobleCircular (la clase ListaEnlazada y su impresión por consola). Tendremos que retocar todo lo necesario para que, ahora, el campo "elemento" de cada nodo sea de tipo "Ruta" en vez de "int". Habrá que retocar "buscar()" para que se busque el arco que contenga el nombre del aeropuerto destino/origen. Habrá también que retocar el método "imprimirListaEnlazada()" para que, por cada elemento de la lista enlazada, se imprima la siguiente terna (mira las pruebas de caja negra): "(aeropuertoOrigenODestino, kms, empresa)".

La clase EspacioAereo tendrá los siguientes atributos:

- Una lista contigua de objetos de tipo Aeropuerto.
- El tamaño del grafo (n).

La clase tendrá también los siguientes métodos privados:

- Método para saber si un aeropuerto existe o no en el espacio aéreo. Se le pasa el nombre del aeropuerto y devuelve true si existe en el grafo, y false si no. Internamente llamará al método "buscar" de la lista contigua.
- Método para obtener la lista de rutas desde un nodo origen. Le pasamos el nombre del aeropuerto origen y nos devuelve el puntero a la lista enlazada de destinos desde ese origen. La precondición es que ese nodo origen exista.
- Método para obtener la lista de rutas hacia un nodo destino. Le pasamos el nombre del aeropuerto destino y nos devuelve el puntero a la lista enlazada de orígenes desde los que se puede llegar a ese destino. La precondición es que ese nodo destino exista.
- Método para saber si hay alguna ruta aérea entre dos aeropuertos. Se le pasa el nombre del aeropuerto origen y el nombre del aeropuerto destino, y el método devuelve true si hay ruta entre ellos o false si no. La precondición es que existan en el grafo ambos aeropuertos.

La clase EspacioAereo tendrá los siguientes métodos públicos (recuerda probar exhaustivamente cada uno antes de empezar a programar el siguiente):

- Constructor (construye un grafo vacío).
- Método para imprimir el espacio aéreo entero. Si no hay nodos, se imprimirá "Grafo vacio". En caso contrario, se imprimirá cada uno de los nodos y, debajo de cada uno de ellos, sangrado con una tabulación, las rutas de salida y las de entrada (ver pruebas de caja negra).
- Método para insertar un aeropuerto cuyo nombre se proporciona. El nodo se inserta desconectado del resto del grafo. Internamente habrá que crear un nuevo Aeropuerto. Fíjate en que, dentro del struct Aeropuerto, tenemos un puntero a un string, no un string propiamente dicho… por lo tanto tendremos que asegurarnos de que creamos el objeto de tipo string antes de asignarlo, para que el constructor de string se ejecute correctamente y el string funcione. Por lo tanto el código tendrá que ser algo similar a esto:

```
void EspacioAereo::insertarAeropuerto(string nuevoNombre) {
	Aeropuerto *nuevoAeropuerto = new Aeropuerto;
	nuevoAeropuerto->nombre = new string;
	*(nuevoAeropuerto->nombre) = nuevoNombre;
	...
}
```

También se podría hacer de esta otra manera:

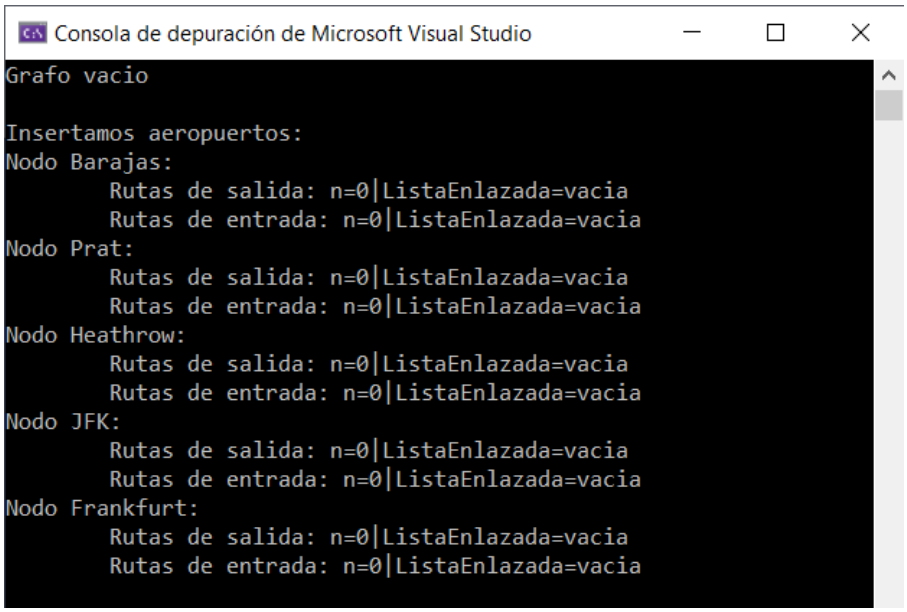
```
void EspacioAereo::insertarAeropuerto(string nuevoNombre) {
	Aeropuerto *nuevoAeropuerto = new Aeropuerto;
	nuevoAeropuerto->nombre = new string(nuevoNombre);
	...
}
```

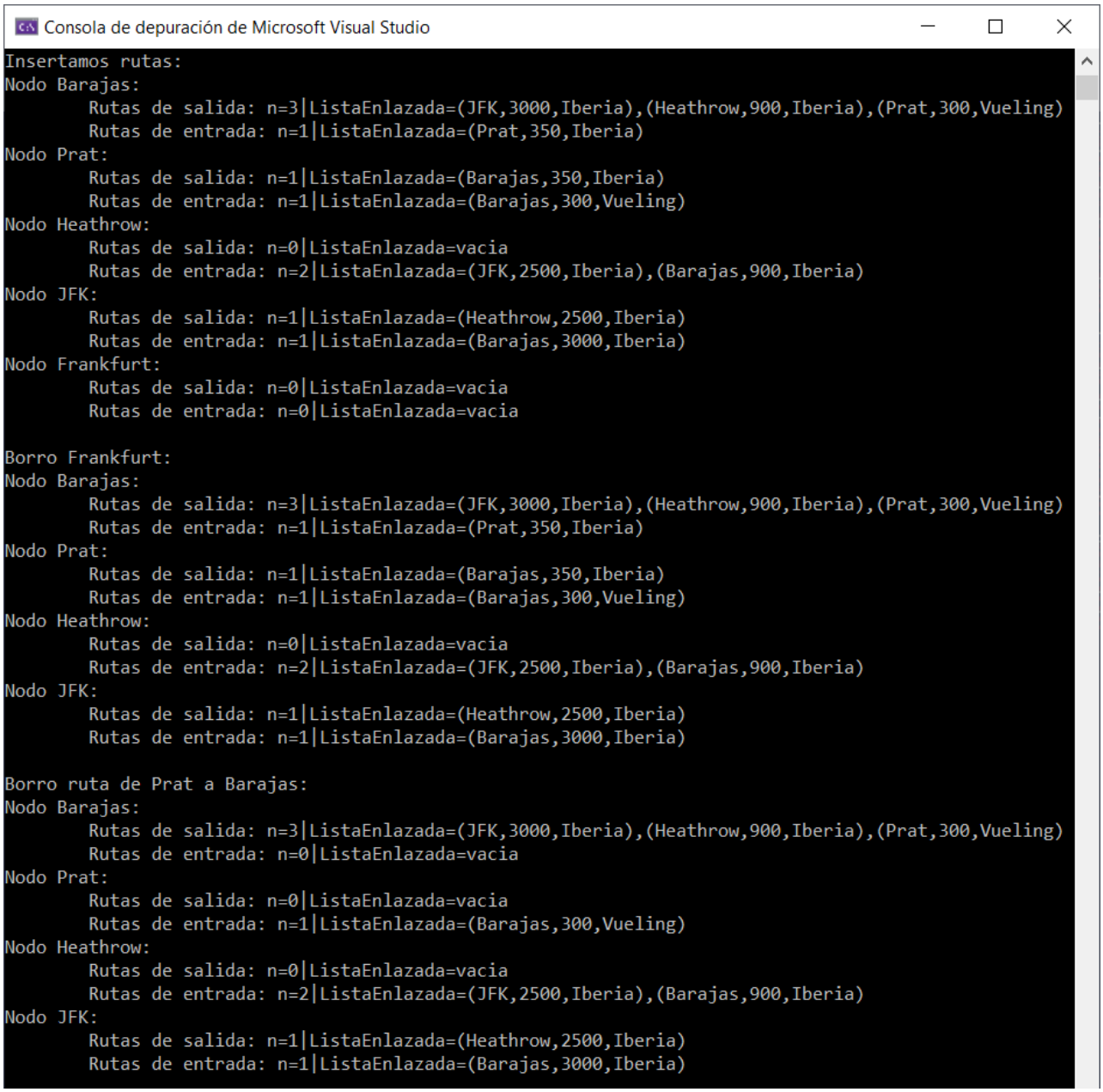
La precondición de este método es que en el grafo no exista ya un nodo con el mismo nombre (usa para ello el método buscar descrito anteriormente).

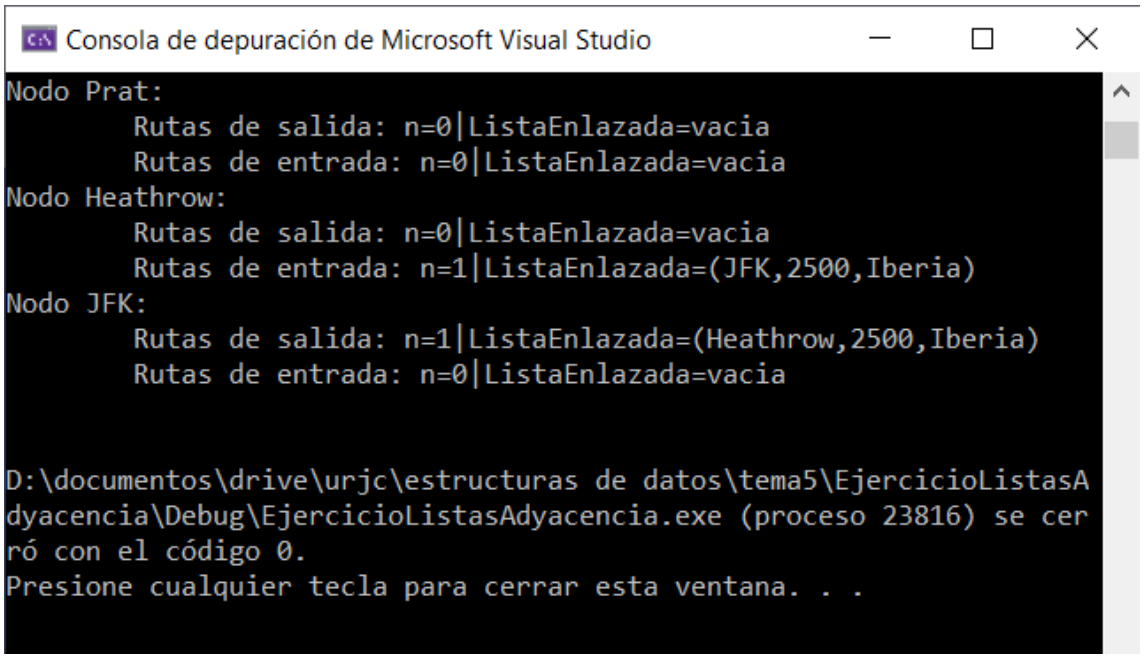
- Método para insertar un arco. Para ello, necesitamos varios parámetros: el nombre del aeropuerto de origen, el nombre del aeropuerto de destino, el numero de kilómetros, y el nombre de la compañía aérea que opera esa ruta. El grafo es dirigido, así pues el orden en que se pasen los dos aeropuertos es relevante. Como el grafo no es un multigrafo, exigimos como precondición que no haya ya en el grafo una ruta entre esos dos aeropuertos. Ej: si intentamos insertar por segunda vez un arco que vaya de "Barajas" a "Prat", la precondición del método no nos dejará. Sin embargo si podremos insertar de "Prat" a "Barajas", pues, al ser el grafo dirigido, se trata de un arco diferente. También exigiremos que ambos aeropuertos existan previamente en el grafo. A la hora de insertar un arco en ambas listas enlazadas, lo insertaremos siempre al principio de dichas listas enlazadas, para tardar lo menos posible (el orden de los arcos en la lista enlazada nos da igual).
- Método para eliminar una ruta. Se la pasa los nombres de los aeropuertos origen y destino. Tanto los aeropuertos como la ruta tienen que existir previamente en el grafo. Al igual que en insertar una ruta, no es lo mismo un arco que va de "Barajas" a "Prat" que de "Prat" a "Barajas". Recordemos que la ruta va a aparecer en dos listas, por lo tanto hay que eliminarla de ambas.
- Método para eliminar un aeropuerto, cuyo nombre se pasa (tiene que existir previamente en el grafo). Se eliminan automáticamente todos los arcos que partan o que lleguen a ese aeropuerto, actualizando todas las listas enlazadas.
- Para no alargar excesivamente este ejercicio, no es necesario implementar el destructor del grafo.

Añade al proyecto un "main" que realice al menos lo siguiente (te sugiero que pruebes muchos más casos, especialmente eliminar nodos y arcos y luego volver a añadir esos mismos nodos y arcos):

1. Crea el espacio aéreo vacío. Imprímelo.
2. Inserta 4 aeropuertos: Barajas, Prat, Heathrow, JFK y Frankfurt. Imprime el grafo.
3. Introduce las siguientes rutas aéreas y después imprime el grafo:
    a. De Barajas a Prat. 300 kms. Operado por Vueling.
    b. De Prat a Barajas. 350 kms. Operado por Iberia.
    c. De Barajas a Heathrow. 900 kms. Operado por Iberia.
    d. De Barajas a JFK. 3000 kms. Operado por Iberia.
    e. De JFK a Heatrow. 2500 kms. Operado por Iberia.
4. Borra Frankfurt. Imprime el grafo.
5. Borra la ruta de Prat a Barajas.
6. Borra Barajas. Imprime el grafo.


Consola de depuración de Microsoft Visual Studio
Grafo vacio
Insertamos aeropuertos:
Nodo Barajas:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo Prat:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo Heathrow:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo JFK:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo Frankfurt:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia


Consola de depuración de Microsoft Visual Studio
Insertamos rutas:
Nodo Barajas:
Rutas de salida: n=3|ListaEnlazada=(JFK,3000,Iberia),(Heathrow,900,Iberia),(Prat,300,Vueling)
Rutas de entrada: n=1|ListaEnlazada=(Prat,350,Iberia)
Nodo Prat:
Rutas de salida: n=1|ListaEnlazada=(Barajas,350,Iberia)
Rutas de entrada: n=1|ListaEnlazada=(Barajas,300,Vueling)
Nodo Heathrow:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=2|ListaEnlazada=(JFK,2500,Iberia),(Barajas,900,Iberia)
Nodo JFK:
Rutas de salida: n=1|ListaEnlazada=(Heathrow,2500,Iberia)
Rutas de entrada: n=1|ListaEnlazada=(Barajas,3000,Iberia)
Nodo Frankfurt:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Borro Frankfurt:
Nodo Barajas:
Rutas de salida: n=3|ListaEnlazada=(JFK,3000,Iberia),(Heathrow,900,Iberia),(Prat,300,Vueling)
Rutas de entrada: n=1|ListaEnlazada=(Prat,350,Iberia)
Nodo Prat:
Rutas de salida: n=1|ListaEnlazada=(Barajas,350,Iberia)
Rutas de entrada: n=1|ListaEnlazada=(Barajas,300,Vueling)
Nodo Heathrow:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=2|ListaEnlazada=(JFK,2500,Iberia),(Barajas,900,Iberia)
Nodo JFK:
Rutas de salida: n=1|ListaEnlazada=(Heathrow,2500,Iberia)
Rutas de entrada: n=1|ListaEnlazada=(Barajas,3000,Iberia)
Borro ruta de Prat a Barajas:
Nodo Barajas:
Rutas de salida: n=3|ListaEnlazada=(JFK,3000,Iberia),(Heathrow,900,Iberia),(Prat,300,Vueling)
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo Prat:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=1|ListaEnlazada=(Barajas,300,Vueling)
Nodo Heathrow:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=2|ListaEnlazada=(JFK,2500,Iberia),(Barajas,900,Iberia)
Nodo JFK:
Rutas de salida: n=1|ListaEnlazada=(Heathrow,2500,Iberia)
Rutas de entrada: n=1|ListaEnlazada=(Barajas,3000,Iberia)


Consola de depuración de Microsoft Visual Studio
Nodo Prat:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=0|ListaEnlazada=vacia
Nodo Heathrow:
Rutas de salida: n=0|ListaEnlazada=vacia
Rutas de entrada: n=1|ListaEnlazada=(JFK,2500,Iberia)
Nodo JFK:
Rutas de salida: n=1|ListaEnlazada=(Heathrow,2500,Iberia)
Rutas de entrada: n=0|ListaEnlazada=vacia
D:\documentos\drive\urjc\estructuras de datos\tema5\EjercicioListasAdyacencia\Debug\EjercicioListasAdyacencia.exe (proceso 23816) se cerró con el código 0.
Presione cualquier tecla para cerrar esta ventana. . .