HOJA N°

FECHA

```
Public class Parcial {
    public ListaGenerica<Ciudad> resolver(Grafo<Ciudad> ciudades,
        String origen, String destino){
        boolean[] marca = new boolean[ciudades.listaDeVertices().tamanio()];
        ListaGenerica<Ciudad> camino = new ListaGenerica<Ciudad>();
        ListaGenerica<Vertice<Ciudad>> = ciudades.listaDeVertices();
        boolean ok = false;
        Vertice<Ciudad> v = null;
        aux.comenzar();
        while(!aux.fin() && !ok){
            v = aux.proximo();
            if(v.dato().nombre().equals(origen)){
                ok = true;
            }
        }
        if(ok && v.dato().case()!=1){
            resolver(ciudades, destino, marca, camino, v.posicion);
        }
        return camino;
    }
```

```
private boolean resolver (Grafo<Ciudad> ciudades, String destino,
boolean [] marca, ListaGenerica<Ciudad> camino, int i){
    boolean encontre = false;
    marca[i] = true;
    Vertice<Ciudad> v = ciudades.listaDeVertices.elementoEn(i);
    camino.agregarFinal(v.dato());
    if (v.dato().nombre().equals(destino){
        encontre = true;
    else {
        ListaGenerica<Arista<Ciudad>> ady = ciudades.listaDeAdyacentes(v);
        ady.comenzar();
        while (!ady.fin() && !encontre){
            Arista<Ciudad> a = ady.proximo();
            int j = a.verticeDestino().posicion();
            if (!marca[j] && a.verticeDestino().dato().compareTo(-1){
                encontre = resolver(ciudades, destino, marca, camino, j);
            }
        }
        if (!encontre){
            marca[i] = false;
            camino.eliminarEn(camino.tamanio());
        }
    }
    return encontre;
}
```