```
Public class Parcial () {

    Public ListaGenerica<ListaGenerica<String>> resolver(Grafo<String> g, lados,
                                                         String origen, String Destino,
    ListaGenerica<ListaGenerica<String>> caminos = new LEG();  String Palabra){
    IF (lc vacia.esVacia()) {
        int pos = Buscar (ciudades, origen, destino);
        if (Pos <> -1) {
            ListaGenerica<String> camActual = new ListaEnlazadaGenerica(),
            Boolean marca = new boolean[ciudades.listaDeVertices.tamaño()];
            dFS(pos, ciudades, caminos, camActual, destino, Palabra,
                                                    marca);
        }
    }
    return caminos;
}

Private void dfs(int pos, Grafo<String> g, LG<LG<String>> caminos,
                                LG<String> camActual, String destino, string Palabra,
    marca[Pos] = true;                                  Boolean[] marca) {
    Vertice<String> vActual = g.vertice(pos);
    camActual.agregarFinal(vActual.dato());
    IF (vActual.dato().equals(destino) && (camActual.incluye(Palabra)){
        caminos.agregarFinal(camActual.clonar());
    }
    else {

        LG<Arista<String>> ady = g.listaDeAdyacencias(vActual);
        ady.comenzar();
        while (! ady.fin()){
            Arista<String> aris = ady.proximo();
            if (aris.Peso.equals("H")){
                int j = aris.verticeDestino().posicion();
                if(! marca[j]) {
                    dFS(j, g, caminos, camActual, destino, Palabra, marca)
                }
            }
        }
    }
    marca[Pos] = False
    camActual.eliminarEn(camActual.tamaño()),
}
```