Parcial 5

```
Public ListaGenerica<Persona> resolver(Grafo<Persona> grafo, Persona
  empleado, int grado) {
  Vertice<Persona> origen = obtenerVertice(empleado, grafo);
  if (grafo.esVacio()) return null;
  if (origen == null) return null;
  boolean[] marca = new boolean[grafo.listaDeVertices().tamanio()];
  ListaGenerica<Persona> resultado;
  resultado = new ListaGenericaEnlazada<Persona>();
  resolver(grafo, origen, grado, marca, resultado);
  return resultado;
}

Private Vertice<Persona> obtenerVertice(Persona persona,
  Grafo<Persona> grafo) {
  ListaGenerica<Vertice<Persona>> vertices = grafo.listaDeVertices();
  Vertice<Persona> actual;
  vertices.comenzar();
  while(!vertices.fin()) {
    actual = vertices.proximo();
    if (actual.dato().equals(Persona)) return actual; }
  return null; }
```

NOTA

```
Private void resolver (Grafo<Persona> grafo, Vertice<Persona> origen,
  int grado, boolean[] marca, Lista Generica<Persona> resultado) {
  Cola Generica<Vertice<Persona>> cola;
  cola = new Cola Genérica<Vértice<Persona>> ();
  int cant = 0;
  Vertice<Persona> actual = origen;
  cola.encolar(actual);
  cola.encolar(null);
  while (!cola.esVacia() && grado >= 0) {
    actual = cola.desencolar(); marca[actual.Posición()] = true;
    if(actual == null) {
      grado--; cola.encolar(null); }
    else {
      if(cant < HC && actual.dato.isJubilado()) {
        resultado.agregarFinal(actual);
        cant++;
        Lista Generica<Arista<Persona>> ady;
        ady = grafo.listaDeAdyacentes(actual);
        ady.comenzar();
        Vertice<Persona> auxiliar;
        while(!ady.fin())
          auxiliar = ady.Proximo().verticeDestino();
          if(!marca[auxiliar.Posicion()] && auxiliar.dato().
            isJubilado)
              cola.encolar(auxiliar);
      }
    }
  }
}
```

```
Public class Persona {
  Private boolean jubilado;
  Private String nombre;
  Private String domicilio;
  Public boolean isJubilado () {
    return jubilado;
  }
}
```