

```

public class BuscadorDeCaminos {

    private Grafo <String> bosque;

    public ListaGenerica < ListaGenerica <String>> recorridosMasSeguros (){

        ListaGenerica<ListaGenerica<String>> resultado =
            new ListaGenericaEnlazada< ListaGenericaEnlazada <String>>();

        ListaGenerica< <String> caminoActual = new ListaGenericaEnlazada <String>();

        boolean [] marca = new boolean [bosque.listaDeVertices().tamanio()+1];

        // Búsqueda de la casa de Caperucita Roja

        Vertice <String> v = buscarCasaCaperucita ();

        caminoActual.agregarFinal (v.dato());

        marca [v.posicion()] = true;

        dfs(v, caminoActual, marca, resultado);

        return resultado;
    }

    private void dfs (Vertice<String> v, ListaGenerica<String> caminoActual, boolean[] marca,
                    ListaGenerica<ListaGenerica<String>> resultado) {

        ListaGenerica<Arista<int>> ady = bosque.listaDeAdyacentes(v);
        ady.comenzar();

        while ( !ady.fin()) {

            Arista a = ady.proximo();
            Vertice <String> vDest = a.verticeDestino();
            int posDest = vDest.posicion();

            if (! marca [posDest] && a.peso() <5) {

                marca [posDest] = true;
                caminoActual.agregarFinal (vDest.dato());

                if (vDest.dato().equals ("Casa Abuelita"))

                    resultado.agregarFinal(caminoActual.copia());

                else

                    dfs (vDest, caminoActual, marca, resultado);

                marca [posDest] = false;

                caminoActual.eliminarEn(caminoActual.tamanio());

            } // fin del 1er If
        } // fin del while
    }
}

```

```
private Vertice<String> buscarCasaCaperucita (){  
  
    ListaGenerica<Vertice<String> >vertices = bosque.listaDeVertices ();  
    vertices.comenzar();  
  
    while(!vertices.fin()){  
  
        Vertice<String> v = vertices.proximo();  
        If (v.dato().equals ("Casa Caperucita"))  
            return v;  
  
    }  
}  
}
```