Ejercicio 2

5.-

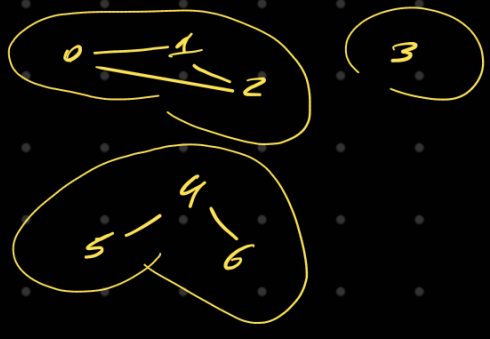
	T	T	T	T	
0	1	2	3	4	5

	0	1	2	3	4
0					
1					
2					
3					
4					

mostrarAmigos(Cliente cliente){

↑ se miran las aristas que hagan de puente y si la hay es que es cercano y por tanto amigos

0	1	2	3	4	5	6




```
public int contarGrupos(){
```

si hago un recorrido en profundidad veo desde el 0 al numVertices-1 sus conexiones

```
for(int i=0; i<numClientes; i++){
    if(!visitados[i]){ ← No visitados
        boolean [] grupo = red.amplitudDesdeVertice(i);
        for(int j=0; j<numClientes; j++){
            if(grupo[j]){ ← Vértices recorridos
                visitados[j] = true; ← No se recorren de nuevo.
            }
        }
        print(clientes[i]);
        numGrupos++;
    }
}
```

Mirar en amplitud y contar personas

```
public ArrayList<Cliente> mayorGrupo() {
    for(int i=0; i<numClientes; i++){
        if(!visitados[i]){
            boolean[] grupo = red.amplitudDesdeVertice(i);
            ArrayList<Cliente> miembrosGrupo = new ArrayList<>();
            for(int j=0; j<numClientes; j++){
                if(grupo[j]){
                    visitados[j] = true;
                    miembrosGrupo.add(clientes[j]);
                }
            }
            if(miembrosGrupo.size() > maxTamano){
                maxTamano = miembrosGrupo.size();
                grupoMasGrande = miembrosGrupo;
            }
        }
    }
}
```

Se recorren los grupos guardando los integrantes

Si el tamaño del grupo es mayor al que se había guardado se reemplaza.

```
public String bebidaGrupo(Cliente cliente){
    int indice = getIndice(cliente);
    String bebidaPredominante = "";
    if(indice != -1){
        boolean[] grupo = red.amplitudDesdeVertice(indice);
        TreeMap<String, Integer> contador = new TreeMap<>();
        for(int i=0; i < numClientes; i++){
            if(grupo[i]){
                String bebida = clientes[i].getBebidaFavorita();
                if(contador.containsKey(bebida)){
                    contador.put(bebida, contador.get(bebida)+1);
                }else{
                    contador.put(bebida, 1);
                }
            }
        }
        int max = 0;
        Iterator<String> it = contador.keySet().iterator();
        while(it.hasNext()){
            String bebida = it.next();
            int cantidad = contador.get(bebida);
            if(cantidad > max){
                max = cantidad;
                bebidaPredominante = bebida;
            }
        }
    }
    return bebidaPredominante;
}
```