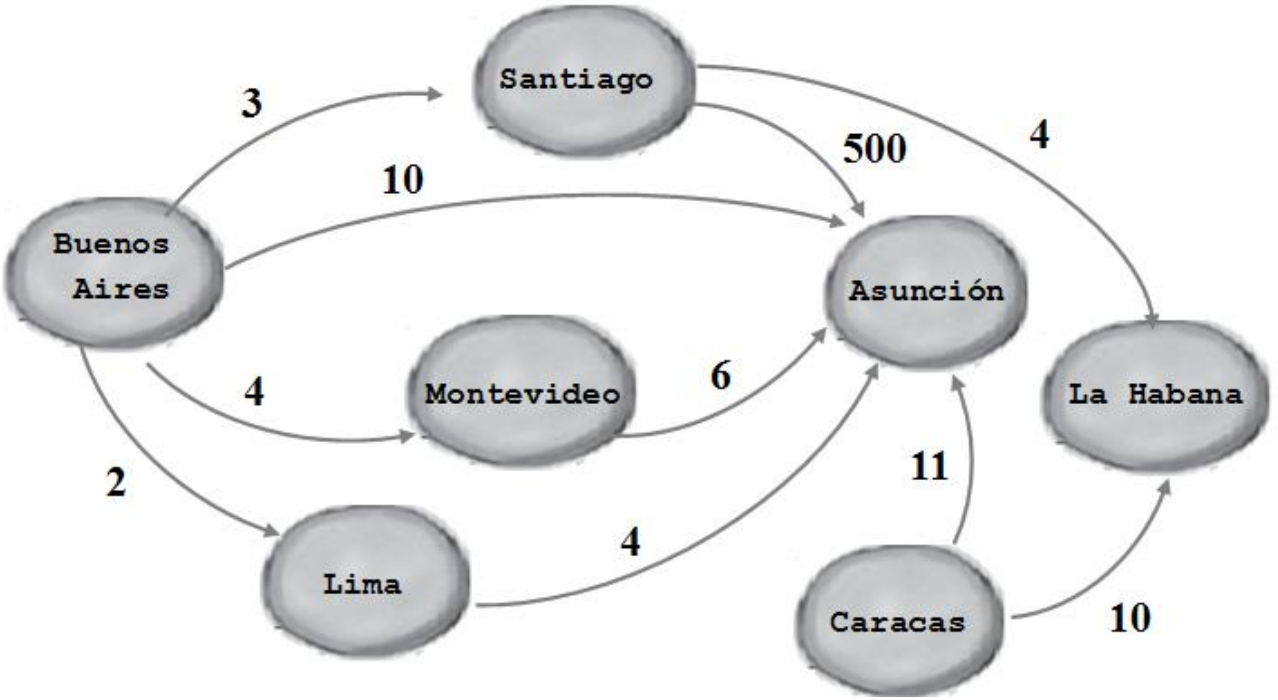
Ejercicio de Parcial

Dado un Grafo orientado y valorado positivamente, como por ejemplo el que muestra la figura, implemente un método que retorne una lista con todos los caminos cuyo costo total sea igual a 10. Se considera ***costo total del camino*** a la suma de los costos de las aristas que forman parte del camino, desde un vértice origen a un vértice destino.

Se recomienda implementar un método público que invoque a un método recursivo privado.



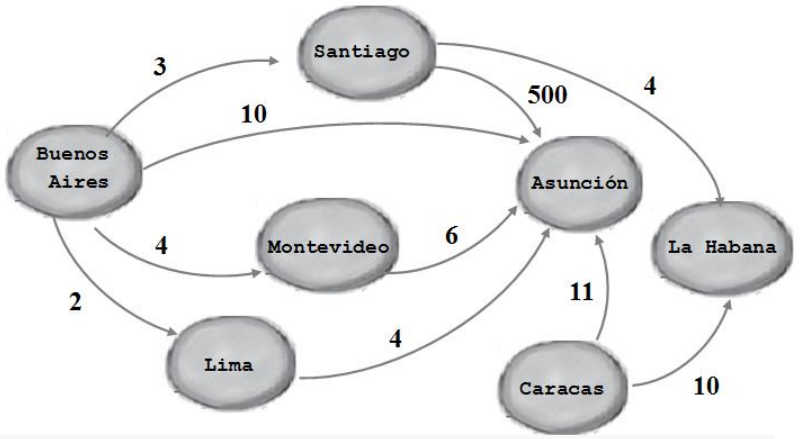
Ejercicio de Parcial (1/2)

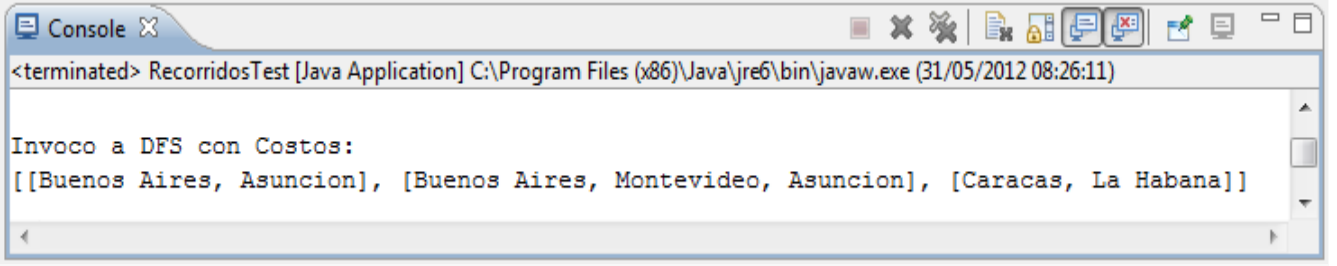
```
public class Recorridos {
  public ListaGenerica<ListaGenerica<Vertice<T>>> dfsConCosto(Grafo<T> grafo){
   boolean[] marca = new boolean[grafo.listaDeVertices().tamanio()+1];
   ListaGenerica<Vertice<T>> lis = null;
   ListaGenerica<ListaGenerica<Vertice<T>>> recorridos =
                              new ListaGenericaEnlazada<ListaGenericaEnlazada<Vertice<T>>>();
   int costo = 0;
   for(int i=1; i<=grafo.listaDeVertices().tamanio();i++){
                    lis = new ListaGenericaEnlazada<Vertice<T>>();
          lis.add(grafo.listaDeVertices().elemento(i));
               marca[i]=true;
          this.dfsConCosto(i, grafo, lis, marca, costo, recorridos);
          marca[i]=false;
   }
   return recorridos;
 }

 private void dfsConCosto(int i, Grafo<T> grafo, ListaGenerica<Vertice<T>> lis,
          boolean[] marca, int costo, ListaGenerica<ListaGenerica<Vertice<T>>> recorridos) {
  ...
 }
}
```

Ejercicio de Parcial (2/2)

```
public class Recorridos {
 private void dfsConCosto(int i, Grafo<T> grafo, ListaGenerica<Vertice<T>> lis,
         boolean[] marca, int costo, ListaGenerica<ListaGenerica<Vertice<T>>> recorridos) {
  Vertice<T> vDestino = null; int p=0,j=0;
  Vertice<T> v = grafo.listaDeVertices().elemento(i);
  ListaGenerica<Arista<T>> ady = grafo.listaDeAdyacentes(v);
  ady.comenzar();
   while(!ady.fin()){
      Arista<T> arista = ady.proximo();
      j = arista.getVerticeDestino().getPosicion();
      if(!marca[j]){
         p = arista.getPeso();
         if ((costo+p) <= 10) {
            vDestino = arista.getVerticeDestino();
            lis.agregarFinal(vDestino);
            marca[j] = true;
            if ((costo+p)==10)
              recorridos.add(lis.copia());
            else
              this.dfsConCosto(j, grafo, lis, marca, costo+p, recorridos);
            lis.eliminar(vDestino);
            marca[j]= false;
          }
       }
      }
  }
}
```





Ejercicio de Parcial

Tiempo de infección de una red

Un poderoso e inteligente virus de computadora infecta cualquier computadora en 1 minuto, logrando infectar toda la red de una empresa con cientos de computadoras. Dado un grafo que representa las conexiones entre las computadoras de la empresa, y una computadora ya infectada, escriba un programa en Java que permita determinar el tiempo que demora el virus en infectar el resto de las computadoras. Asuma que todas las computadoras pueden ser infectadas, no todas las computadoras tienen conexión directa entre sí, y un mismo virus puede infectar un grupo de computadoras al mismo tiempo sin importar la cantidad.

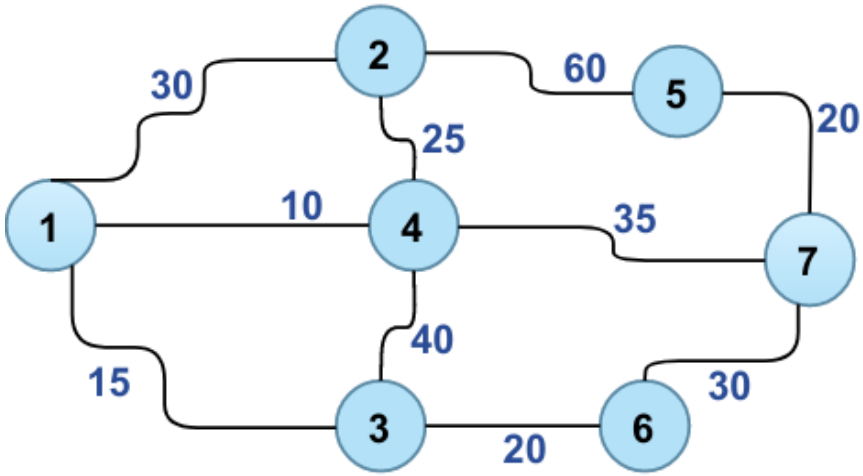
Ejercicio de Parcial

```java
public class BFSVirusNull {
    public static int calcularTiempoInfeccion(Grafo<String> g, Vertice<String> inicial) {
        int tiempo = 0;
        boolean visitados[] = new boolean[g.listaDeVertices().tamanio()+1];
        ColaGenerica<Vertice<String>> cola = new ColaGenerica<Vertice<String>>();
        visitados[inicial.getPosicion()] = true;
        cola.encolar(inicial);
        cola.encolar(null);
        while (!cola.esVacia()) {
            Vertice<String> v = cola.desencolar();
            if (v != null) {
                ListaGenerica<Arista<String>> adyacentes = v.obtenerAdyacentes();
                adyacentes.comenzar();
                while (!adyacentes.fin()) {
                    Arista<String> a = adyacentes.proximo();
                    Vertice<String> w = a.getVerticeDestino();
                    if (!visitados[w.getPosicion()]) {
                        visitados[w.getPosicion()] = true;
                        cola.encolar(w);
                    }
                }
            } else if (!cola.esVacia()) {
                tiempo++;
                cola.encolar(null);
            }
        }
        return tiempo;
    }
}
```

Para pensar ...

Problema: El Guía de Turismo

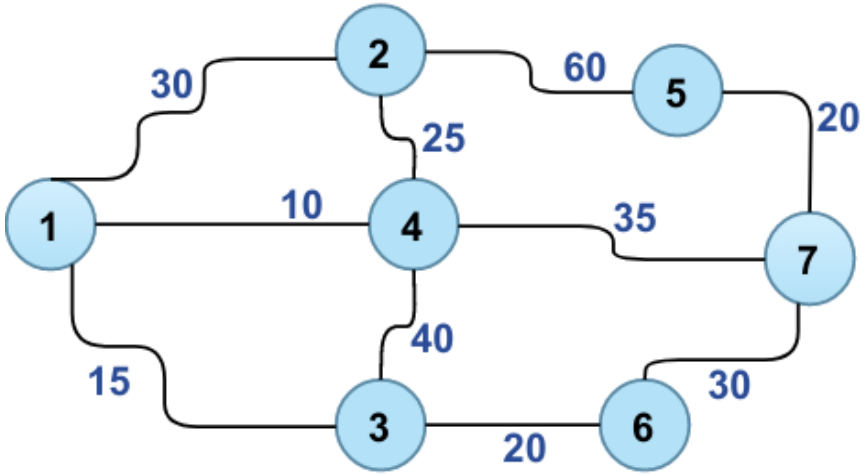
El señor H es un guía de turismo de la ciudad de Buenos Aires. Su trabajo consiste en mostrar a grupos de turistas diferentes **puntos de interés** de la ciudad.
Estos puntos de interés están **conectados por rutas en ambos sentidos**. Dos puntos de interés vecinos tienen un servicio de bus que los conecta, con una limitación en el **número máximo de pasajeros** que puede transportar. No es siempre posible para el señor H transportar de una única vez a todos los turistas a un destino en particular.



Por ejemplo, consideremos el siguiente mapa con **7 puntos de interés**, donde las aristas representan las rutas y **el peso de ellas representa el límite máximo de pasajeros a transportar por el servicio de bus**. Su misión es indicarle al Sr. H cuál es el menor número de viajes que deberá realizar para llevar al grupo de turistas de un origen a un destino.

Problema: El Guía de Turismo

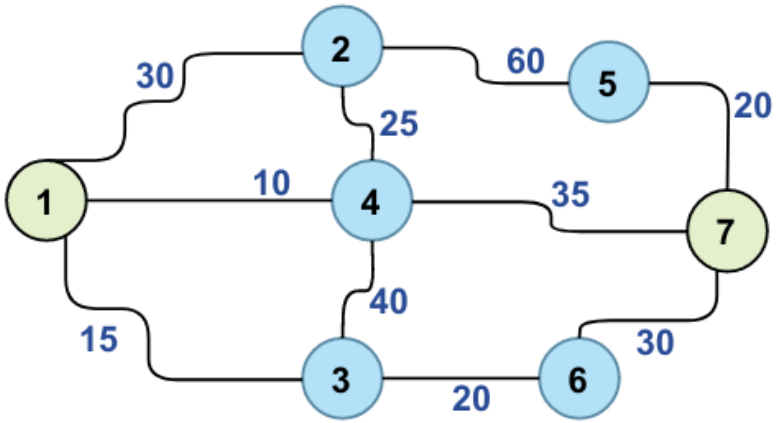
El señor H es un guía de turismo de la ciudad de Buenos Aires. Su trabajo consiste en mostrar a grupos de turistas diferentes **puntos de interés** de la ciudad.
Estos puntos de interés están **conectados por rutas en ambos sentidos**. Dos puntos de interés vecinos tienen un servicio de bus que los conecta, con una limitación en el **número máximo de pasajeros** que puede transportar. No es siempre posible para el señor H transportar de una única vez a todos los turistas a un destino en particular.



Por ejemplo, consideremos el siguiente mapa con **7 puntos de interés**, donde las aristas representan las rutas y **el peso de ellas representa el límite máximo de pasajeros a transportar por el servicio de bus**. Su misión es indicarle al Sr. H qué camino debe tomar para llevar un grupo de turistas de un origen a un destino y haciendo el menor número de viajes.

Supongamos que el señor H debe transportar a **99 turistas** del punto **1** al punto **7**.
Cuáles son los recorridos posibles? Cuál implica realizar el menor número de viajes?

Problema: El Guía de Turismo



Puntos de interés	Cant. turistas x viaje	Cant. viajes
1 2 4 3 6 7	20	6
1 2 4 7	**25**	**5**
1 2 5 7	20	6
1 3 4 2 5 7	15	8
1 3 4 7	15	8
1 3 6 7	15	8
1 4 2 5 7	10	11
1 4 3 6 7	10	11
1 4 7	10	11

Problema: Entregar sueldo a jubilados

El **Banco Itaú** se suma a las campañas "**QUEDATE EN CASA**" lanzando un programa para acercarle a los jubilados el sueldo hasta sus domicilios. Para ello el banco cuenta con información que permite definir un grafo de personas donde la persona puede ser un jubilado o un empleado del banco que llevará el dinero.
Se necesita armar la cartera de jubilados para que cada empleado repartidor del banco, incluyendo en cada lista los jubilados que vivan un radio cercano a su casa y no hayan percibido la jubilación del mes.
Para ello, implemente un algoritmo que dado un **Grafo<Persona>** retorne una lista de jubilados que se encuentren a una distancia menor a un valor dado del empleado Itaú (grado de separación del empleado Itaú).
El método recibirá un **Grafo<Persona>**, **un empleado** y un **grado de separación/distancia** y debe retornar una lista de hasta 40 jubilados que no hayan percibido la jubilación del mes y se encuentres a una distancia menor a recibido como parámetro.

En este grafo simple, donde los empleados del banco están en color rojo y se desea retornar los jubilados hasta distancia 2, se debería retornar los los jubilados color negro.
La **Persona** conoce si es empleado o jubilado, el nombre, domicilio.