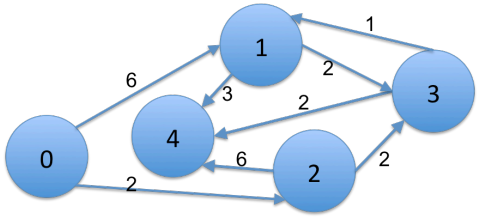
**Corso di Algoritmi 2 - Prova pratica di programmazione**

Un piccolo produttore di torte gelato che consegna a domicilio (e non vuole investire in un camion frigorifero!) decide di usare un software per stabilire la quantità di ghiaccio secco minimo che serve per consegnare ciascuna torta.
Ha accesso a una mappa della città in formato digitale in cui in ogni momento sono disponibili i tempi di percorrenza (in dipendenza dal traffico) per ciascuna tratta percorribile. Serve un software per stabilire la quantità di ghiaccio secco necessaria per imballare ciascuna spedizione, tenuto conto che serve *s* ghiaccio secco per unità di tempo e per unità di peso di torta (cioè se la torta pesa 2 e deve viaggiare per un tempo pari a 10, servirà 20**s* ghiaccio secco). Il parametro *s* varia in base al clima e ad altri fattori.

Scrivere una classe Java **TorteGelato** con il costruttore:
**public TorteGelato (DirectedGraph *mappa, int base, int parametroGhiaccio*),** dove *mappa* è il grafo orientato, pesato, in cui i nodi rappresentano dei punti nella città e il peso dell'arco (*u*,*v*) rappresenta la distanza (in termini di tempo) tra i due punti *u* e *v; base* è il luogo in cui vengono prodotte e da cui vengono spedite le torte gelato; *parametroGhiaccio* è un parametro che indica la quantità di ghiaccio secco necessaria per unità di tempo e unità di peso della torta.
e il metodo:
**public int ghiaccioMinimo (int *dest*, int *weight*)**
che restituisce il minimo ghiaccio secco necessario per imballare la spedizione fino alla destinazione *dest* di una torta di peso *weight*; restituisce -1 se la destinazione non è raggiungibile; lancia un'eccezione java.lang.IllegalArgumentException se *dest* non è una destinazione ammissibile o *weight* non è un numero positivo.




**Esempio :** sul grafo riportato qui a fianco con base= 2, parametroGhiaccio = 3
**ghiaccioMinimo(4,3) = 36**
**ghiaccioMinimo(1,5) = 45**
**ghiaccioMinimo(0,2) = -1**
**ghiaccioMinimo(7,1) throws IllegalArgumentException**
**ghiaccioMinimo(3,-5) throws IllegalArgumentException**

**TEST:** Lavorate implementando anche una classe test JUnit con almeno due test **significativi**, va bene anche se sono semplici. La classe test si deve chiamare **TestTorteGelato.**

**Struttura del progetto Java e consegna:**
Le classi **TorteGelato** e **TestTorteGelato** devono essere contenute in un package il cui nome è (il vostro) **nome.cognome** in minuscolo. Consegnate l'intero package.

**Suggerimenti e commenti:**
1) Potete usare la libreria graphLib.jar.
2) Potete aggiungere tutti i metodi private che volete.
3) Vi ricordo che la sintassi per la descrizione di un grafo nella libreria graphLib è del tipo: "3; 0 1 4; 1 2 5" dove 3 è il numero dei nodi, (0,1) e (1,2) sono archi di peso 4 e 5 rispettivamente.
4) Vi ricordo che il codice per lanciare l'eccezione è: *throw new IllegalArgumentException("messaggio").*
5) Suggerimento: Selezionate la cartella "Consegna" come vostro workspace; in questo modo, in caso di problemi tecnici, il vostro lavoro non andrà completamente perduto, e comunque non rischiate di dimenticarvi di consegnare (se invece doveste decidere di ritirarvi, basta dirlo).
6) Verranno valutatati: l'algoritmo utilizzato, la correttezza e l'aderenza alle specifiche (comprese la gestione degli input particolari e l'inizializzazione), e la presenza di una classe test con test significativi.