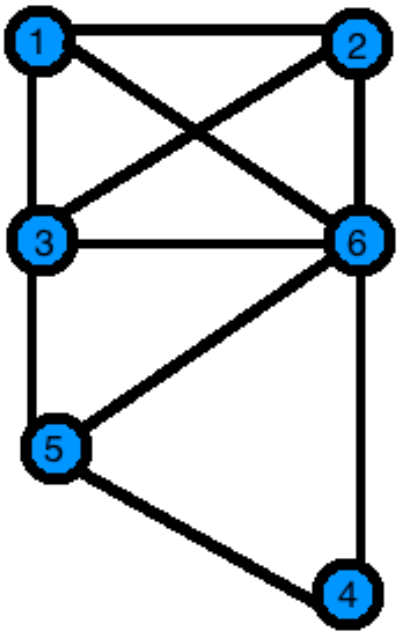
COURS_4 : LE PLUS COURT CHEMIN DANS UN GRAPHE 1

I. Problème Simple (graphe non pondéré) :

I : G = (V,E) s∈V , t ∈V (s->sommet debut et t -> sommet source)
S : {path(s,t)$_g$}
f : lpath(s,t)$_g$l
opt : min

Les distances = 1



s = 2
t = 4

Père	2	2	2	2	2	2

Comment utiliser ***l'algorithm noyeau*** dans notre cas ?

Algorithm Nouyeau :

```
Init P, M..
add(Q,u)

while Q :
        z = choose_a_member_of_Q
        for each x in Adj[x] :
                PREVISIT_x
                if (M[x] == 0) :
                        M[x] = 1
                        VISIT_x
                        add(Q,x)
        POSTVISIT_z
```

-> M = [0 for un in G]
-> Niv = [0 for un in G]
-> P = [-1 for un in G]

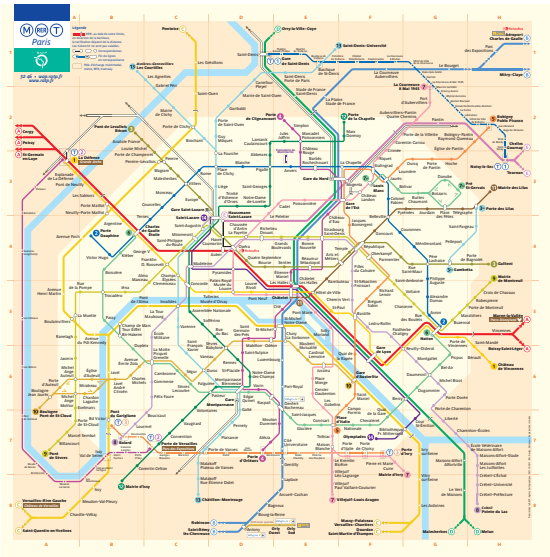
-> z= Q.dequeue()

-> P[x] = z
-> Q.enqueue(x)
-> Niv[x] = Niv[z] + 1

une fonction pour tracer ensuite le chemin :

```
def chemin(P, s, t) :
        R = []
        u = t
        while u != s :
                R.insert(Q,u)
                u = P[u]
        return R.insert(Q,u)
```

II. EXERCICE : LE MINIMUM DE CORRESPONDANTE :

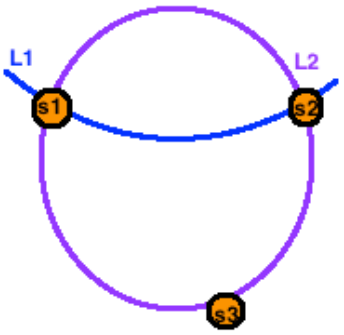


FORMALISATION :

I : $L = \{L_1, L_2, L_3, L_4, ...\}$ s (station départ) , t (station de fin)

S : $\{(s_1, s_2, s_3, ... s_K / \forall i\ 1 \leq i \leq k-2\ \exists \lambda : s_i \in L_\lambda , s_{i+1} \in L_\lambda, s_{i+2} \in L_\lambda \}$

on fait le graphe suivant :
chaque sommet représente une station
chaque arrête reliant un sommet u à un sommet v signifie que u est v sont parcouru par une même ligne

ex : 

donnera en graphe