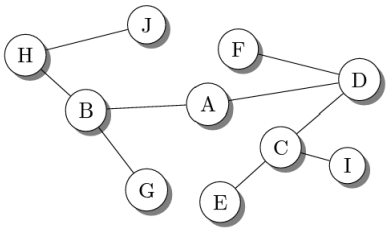
III.6 Samedi 18 avril ♪ ♪ ♪

1–❑ Qu'est-ce qu'une pile ? qu'est-ce qu'une file ? S'aider d'une représentation de la vie quotidienne !

Solution: Une pile est une structure informatique basée sur le principe « dernier arrivé, premier sorti ». Une liste peut être vue comme une pile : `L.append(x)` ajoute `x` en fin de liste, et `L.pop()` enlève le dernier élément de `L`.
Une file est une structure informatique basée sur le principe « premier arrivé, premier sorti. » `L.append(x)` ajoute `x` en fin de file, et `L.popleft()` enlève le premier élément de `L`.



2–❑ Effectuer le parcours en largeur et en profondeur du graphe suivant à partir de A :

Solution:
- Parcours en largeur : A - B - D - C - H - F - C - J - E - I
- Parcours en profondeur : A - B - H - J - D - C - E - G - F - E - I

3–❑ Compléter la fonction suivante qui effectue de façon itérative le parcours en largeur d'un graphe représenté par un dictionnaire l'adjacence.

Solution:
```
def parcours_largeur(G,s0):
    parcours=[] # liste décrivant le parcours
    vu={} # dictionnaire  des sommets visités
    file=deque([s0]) # file qui garde les sommets en attente
    while len(file)>0 : # tant que la file n'est pas vide
        s=file.popleft() # on enlève le sommet au début de la file (file.popleft())
        if s not in vu : # si le sommet s n'a pas déjà été visité
            parcours.append(s) # on l'ajoute à la liste du parcours
            vu[s]=True # on l'ajoute au dictionnaire vu
            for s in G[s] : # il faut visiter les voisins v du sommet s
                if s not in vu : # le voisin v de s n'a pas déjà été visité
                    file.append(s) # on le place dans la file des sommets en
   attente d'être visité (en fin de file)
    return visite
```

4–❑ Compléter la fonction suivante qui effectue de façon itérative le parcours en profondeur d'un graphe représenté par un dictionnaire l'adjacence.

Solution:
```
def parcours_profondeur(G,s0):
    parcours=[] # liste décrivant le parcours
    vu={} # dictionnaire  des sommets visités
    pile=[] # pile qui garde les sommets en attente
    pile.append(s0) # on ajoute au sommet de la pile le sommet de départ
    while len(pile)>0 : # tant que la pile n'est pas vide
        s=pile.pop() # on enlève et enlève le sommet au sommet de la pile (pile.pop
   ())
        if s not in vu : # si le sommet s  n'a pas déjà été visité
            parcours.append(s) # on l'ajoute à la liste des sommets visités
            vu[s]=True # on l'ajoute au dictionnaire des sommets vus
            for v in G[s] : # il faut visiter les voisins v du sommet s
                if v not in vu : # si le voisin v de s n'a pas déjà été visité
                    pile.append(v) # on le place dans la pile des sommets en
   attente d'être visité
    return parcours
```