**Sujet 2**

**Exercice 1 :**

1. Sa taille est de 8.
2. Sa hauteur est de 4.

13

11

14

1. Les valeurs du sous arbre gauche sont plus petites que la racine et les valeurs du sous-arbre droits sont plus grande que la racine.

Prof : Dans l’arbre et ses sous arbres, la valeur d’une racine est inferieure aux valeurs de son arbre gauche et inférieur aux de son arbre droit.

1. La valeur 17 est le sous-arbre gauche du nœuds 18 (voir correction pour voir l’arbre en entier.
2. La lettre correspondant à celle qui construit et stocke dans la variable abr l’arbre représenté ci-contre est la lettre C.
3. return Nœud(ins(v,abr.gauche)), abr.valeur, abr.droit))
4. Quand nous exécutons nb\_sup(16, abr) alors le résultats est 4 mais son nombre d’appel est de 17.

1 + 2 + 2 + 2 +2 + 2 +2 +2 + 2 = 17

1. def nb\_sup(v, abr):

if abr is None:

return 0

else:

if abr.valeur >= v:

return 1 + nb\_sup(v, abr.gauche) + nb\_sup(v, abr.droit)

else:

return nb\_sup(v, abr.droit)

**Exercice 2 :**

1. Voici les différentes étapes :

Etape 1 : 498742 🡪 498742 🡪 4842

Etape 2 : 4842 🡪 442

Etape 3 : 442 🡪 42

1. C’est la pile B qui est gagnante. Voici les détails :

Pile A : 545421 🡪 55421 🡪 55421 🡪 55421 🡪 55421 jusqu’à l’infini

Pile B : 454920 🡪 44920 🡪 4420 🡪 420 🡪 40 🡪 Gagnant

Pile C : 348761 🡪 348761 🡪 348761 🡪 34861 🡪 34861 🡪 3461 🡪 3461 à l’infini

1. def reduire\_triplet\_au\_sommet(p):

a = depiler(p)

b = depiler(p)

c = sommet(p)

if a % 2 != c % 2 :

empiler(p, b)

empiler(p, a)

1. La taille minimale que doit avoir une pile pour être réductible est de 3.
2. def parcourir\_pile\_en\_reduisant(p):

q = creer\_pile\_vide()

while taille(p) >= 3 :

reduire\_triplet\_au\_sommet(p)

e = depiler(p)

empiler(q, e)

while not est\_vide(q):

f = depiler(q)

empiler(p, f)

return p

1. def jouer(p):

q = parcourir\_pile\_en\_reduisant(p)

if q == p :

return p

else:

return jouer(q)

**Exercice 3 :**

1. L’adresse du réseau sur lequel se trouve cette machine est 192.168.0.0
2. L'adresse de diffusion (broadcast) de ce réseau est 192.168.0.255
3. Le nombre maximal de machines que l'on peut connecter sur ce réseau est de 254.

Détails : 256 – 1 (adresse réseau) – 1 ‘adresse broadcast) = 254

1. 192.168.1.48
2. Le paquet sort de la machine 192.168.1.1 puis passe par le switch 192.168.1.0/24 puis passe par le routeur B C F ou C F ou E puis passe par le routeur D puis passe par le switch 192.168.4.0/24 puis arrive à la machine 192.168.4.2.
3. Si une route est en panne, ralenti ou a un problème alors il vaut mieux avoir plusieurs autres routes sinon les paquets envoyés peuvent ne pas s’envoyer.

|  |  |
| --- | --- |
| **Routeur A** | |
| Destination | Passe par |
| B | B |
| C | C |
| D | E |
| E | E |
| F | C |

1. Le parcours emprunté par le paquet est B 🡪 C 🡪 E 🡪D.

|  |  |
| --- | --- |
| **Routeur A** | |
| Destination | Passe par |
| B | B |
| C | C |
| D | C (Puis F) |
| E | C |
| F | C |

|  |  |
| --- | --- |
| **Routeur B** | |
| Destination | Passe par |
| A | A |
| C | C |
| D | C (Puis F) |
| E | C |
| F | C |

|  |  |
| --- | --- |
| **Routeur C** | |
| Destination | Passe par |
| A | A |
| B | B |
| D | E |
| E | E |
| F | F |

1. Le nouveau parcours emprunté par le paquet IP pour aller du routeur B au routeur D est B 🡪 C 🡪 F 🡪 D.

Non c’est B 🡪 A 🡪 C 🡪 E 🡪 D

1. Voici les différents coûts :

Coût Ethernet = 10\*\*8 / 10\*\*7 = 10

Coût Fast-Ethernet = 10\*\*8 / 10\*\*8 = 1

Coût Fibre = 10\*\*8 / 10\*\*9 = 0,1



0,1

0,1

1

0,1

1

0,1

1

10

1. Voici la liste des routes possibles :

1 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur A 🡪 Routeur E 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 22,1 (sans les liaisons switch-routeur : 2,1)

2 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur A 🡪 Routeur C 🡪 Routeur F 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 22,2 (sans les liaisons switch-routeur : 2,2)

3 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur A 🡪 Routeur E 🡪 Routeur C 🡪 Routeur F 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 23,2 (sans les liaisons switch-routeur : 3,2)

4 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur C 🡪 Routeur A 🡪 Routeur E 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 31,2 (sans les liaisons switch-routeur : 11,2)

5 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur C 🡪 Routeur E 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 30,2 (sans les liaisons switch-routeur : 10,2)

6 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur C 🡪 Routeur F 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 31,1 (sans les liaisons switch-routeur : 11,1)

7 : 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur A 🡪 Routeur C 🡪 Routeur E 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1

* Coût : 21,3 (sans les liaisons switch-routeur : 1,3)

1. Pour aller d'une machine ayant pour adresse IPv4 192.168.2.1 à une machine ayant pour adresse IPv4 192.168.4.1, un paquet va prendre la route 7 (soit 192.168.2.1 🡪 192.168.2.0/24 🡪 Routeur B 🡪 Routeur A 🡪 Routeur C 🡪 Routeur E 🡪 Routeur D 🡪 192.168.4.0/24 🡪 192.168.4.1) car c’est celle qui as le coût le faible et donc c’est elle qui va être choisis grâce au protocole OSPF.

**Exercice 4 :**

1. Cette requête affiche tous les titres des morceaux de l’interprète 4.
2. SELECT nom FROM interpretes WHERE pays = « Angleterre »
3. Cette requête affiche le tout titre et les années des titres de chaque morceau dans l’ordre des années.
4. SELECT COUNT(\*) FROM morceaux
5. SELECT titre FROM morceaux ORDER BY titre
6. La clé étrangère de la table morceaux est id\_interprete, car cet attribut permet d’établir un lien avec la table interpretes (il correspond à l’attribut id\_interprete de la table interpretes.
7. Voici mes schémas relationnels :

interpretes(id\_interprete : INT, nom : TEXT, pays : TEXT)

morceaux(id\_morceau : INT, titre : TEXT, annee : INT, #id\_interprete : INT)

1. Cette requête provoque une erreur, car elle essaye d’ajouter à la table interpretes une entrée ayant pour id\_interprete 1. Or, l’attribut id\_interprete (qui est une clé primaire) a déjà une entrée avec la valeur 1 (la clé primaire doit être unique).
2. UPDATE morceaux SET annee = 1971 WHERE titre = 'Imagine'
3. INSERT INTO interpretes(id\_interprete, nom, pays) VALUES(6, 'The Who', 'Angleterre')
4. INSERT INTO morceaux(id\_morceau, titre, annee, id\_interprete) VALUES(7, 'My Generation', 1965, 6)
5. SELECT titre FROM morceaux JOIN interpretes ON interpretes.id\_interprete = morceaux.id\_interprete WHERE pays = 'États-Unis'

**Exercice 5 :**

1. cellule = Cellule(True, False, True, True)
2. Voici les lignes compléter :

Ligne 6 : for i in range(hauteur):

Ligne 8 : for j in range(longueur):

Ligne 9 : cellule = Cellule(True, True, True, True)

Ligne 10 : ligne.append(cellule)

1. cellule2.murs['S'] = False
2. Voici les lignes compléter :

Ligne 19 : cellule2.murs['S'] = False

# cellule2 à l'Ouest de cellule1

Ligne 21 : elif c1\_col - c2\_col == 1 and c1\_lig == c2\_lig:

Ligne 22 : cellule1.murs['O'] = False

Ligne 23 : cellule2.murs['E'] = False

1. Voici les lignes compléter :

Ligne 26 : for k in range(colonne, colonne + long - 1):

Ligne 29 : for k in range(ligne, ligne + haut - 1):

Ligne 30 : self.creer\_passage(k, colonne, k+1, colonne)

