Centres étrangers - mai 2022 - sujet 2

Exercice 1 (Programmation générale et récursivité - 4 points)

1. Voici une fonction Python :

```
def f(n):
    if n == 0:
        print("Partez!")
    else:
        print(n)
        f(n-1)
```

   (a) Qu'affiche la commande `f(5)` ?

   (b) Pourquoi dit-on de cette fonction qu'elle est récursive ?

2. On rappelle qu'en Python l'opérateur + a le comportement suivant sur les chaînes de caractères :

```
>>> S = 'a'+'bc'
>>> S
'abc'
```

   Et le comportement suivant sur les listes :

```
>>> L = ['a'] + ['b', 'c']
>>> L
['a', 'b', 'c']
```

   On a besoin pour les questions suivantes de pouvoir ajouter une chaîne de caractères `s` en préfixe à chaque chaîne de caractères de la liste `liste`. On appellera cette fonction `ajouter`.
   Par exemple, `ajouter("a",["b","c"])` doit renvoyer `["ab","ac"]`.

   (a) Recopier le code suivant et compléter les `.....` sur la copie :

```
def ajouter(s, liste):
    res = []
    for m in liste:
        res .....
    return res
```

   (b) Que renvoie la commande `ajouter("b", ["a","b","c"])` ?

   (c) Que renvoie la commande `ajouter("a", [""])` ?

3. On s'intéresse ici à la fonction suivante écrite en Python où `s` est une chaîne de caractères et `n` un entier naturel.

```
def produit(s, n):
    if n == 0:
        return [""]
    else:
        res = []
        for i in range(len(s)):
            res = res + ajouter(s[i], produit(s, n-1))
        return res
```

   (a) Que renvoie la commande `produit("ab", 0)` ? Le résultat est-il une liste vide ?

   (b) Que renvoie la commande `produit("ab", 1)` ?

   (c) Que renvoie la commande `produit("ab", 2)` ?

Exercice 2 (Dictionnaires - 4 points)
La cryptographie est un ensemble de techniques permettant de chiffrer un message.

Une technique de cryptographie consiste à mélanger les lettres d'un alphabet et à réécrire le message avec ces permutations. En Python, on peut créer un dictionnaire dans lequel les clés sont les lettres de l'alphabet et les valeurs sont celles de l'alphabet mélangé.

Par exemple, si l'alphabet contient les quatre lettres A, B, C et D, et si le dictionnaire de l'alphabet mélangé est

```
alpha = {"A": "B", "B": "D", "C": "A", "D": "C"}
```

alors la chaîne de caractères `"BAC"` sera chiffrée par `"DBA"`.

Un tel dictionnaire sera appelé **dictionnaire de chiffrement**.

1. On souhaite chiffrer un message écrit avec l'alphabet A, B, C, D, E, F, G à l'aide du dictionnaire

   ```
   alpha = {"A":"B", "B":"D", "C":"A", "D":"C", "E":"F", "F":"G", "G":"E"}
   ```

   (a) Quelle est la valeur associée à la clé `"D"` ? En Python, comment l'obtenir ?

   (b) Chiffrer la chaîne de caractères `"BAGAGE"` avec le dictionnaire `alpha`.

2. On considère qu'un mot est une chaîne de caractères (un objet de type `str`) écrite uniquement avec les 26 lettres de l'alphabet en majuscule.
   Par exemple, `"ARBRE"` est un mot et `"L'ARBRE !"` n'est pas un mot à cause des caractères `"'"`, `" "` (espace) et `"!"`.
   Ecrire une fonction `chiffrer` qui prend en paramètres un mot `mot` et un dictionnaire de chiffrement `alpha`, et qui renvoie une chaîne de caractères chiffrée avec le dictionnaire de chiffrement `alpha`.

3. On souhaite déchiffrer un mot chiffré avec cette méthode.

   (a) Si un mot est chiffré avec le dictionnaire de chiffrement

   ```
   alpha = {"A":"B", "B":"D", "C":"A", "D":"C", "E":"F", "F":"G", "G":"E"}
   ```

   donner un dictionnaire permettant de le déchiffrer.

   (b) Ecrire une fonction en Python appelée `dico_dechiffrement` qui prend en paramètre un dictionnaire de chiffrement `dico` et qui renvoie un dictionnaire permettant le déchiffrement. On pourra s'inspirer du code incomplet ci-dessous ou proposer une autre solution :

   ```
   def dico_dechiffrement(dico):
       nouveau = {}
       for lettre in dico :
           code = dico[.....................]
           nouveau[.....................] = .....................
       return nouveau
   ```

   (c) Ecrire une fonction `dechiffre` qui prend en paramètre un mot chiffré `mot` et un dictionnaire de chiffrement `dico` et renvoie le mot décodé. On utilisera les fonctions écrites dans les questions précédentes.

4. On souhaite à présent créer un dictionnaire de chiffrement. Ecrire une fonction `dico_chiffrement` qui prend en paramètre un tableau de lettres `alphabet` et qui renvoie un dictionnaire de chiffrement dont les clés sont les lettres du tableau `alphabet` et les valeurs sont les lettres du tableau `alphabet` mélangées.
   On pourra utiliser la fonction `shuffle` du module `random` qui mélange en place un tableau. Par exemple, on a :

   ```
   >>> tab = ["A", "B", "C", "D"]
   >>> shuffle(tab)
   >>> tab
   ["B", "A", "D", "C"]
   ```

Exercice 3 (Bases de données et SQL - 4 points)

Un rappel sur la syntaxe de quelques fonctions SQL est donné en annexe 1 en fin de sujet.

Les enseignants d'un établissement imaginaire proposent des parcours d'entraînement au numérique à leurs élèves en créant des séries d'exercices appelées **Evaluations**. Les différentes informations sont stockées dans une base de données.

Les informations de chaque campagne créée sont stockées dans la table `Evaluations` dont la structure est la suivante :

attribut	type
Code_evaluation	CHAR
Nom_evaluation	CHAR
Auteur	CHAR
Date	CHAR
Code_competences	INT

Un extrait de la table `Evaluations` est donné ci-dessous :

Code_evaluation	Nom_evaluation	Auteur	Date	Code_competences
EXKVLX886	Term7	Peltier	2021-10-13	1453
AZVBYB689	Groupe3	Lacour	2021-10-07	1276
PRJUYR491	Term5	Peltier	2021-10-07	1453
RTKVLX656	campagneSTMG	Beley	2021-10-03	476
DZLYYR479	Term5	Serhani	2021-09-27	1659
XJVBTX585	grNSI2	Eisen	2021-09-24	532
CRLYYR439	1ere6	Caille	2021-09-13	532
AZVBYB789	rentreeHGGSP	Martin	2021-09-13	386
OBJUYR491	Web_2nde	Boucher	2021-09-07	452
AGTBYB689	rechercheBTS	Beley	2021-09-07	1341
DQVBTX905	2nde2	Nguyen	2021-09-07	452

TABLEAU 1

1. (a) Dans la table `Evaluations`, quel est le seul attribut pouvant servir de clé primaire ? Justifier votre réponse.
   (b) Ecrire la requête SQL d'insertion qui a permis d'enregistrer la campagne `Term7` dans la table `Evaluations`. Les informations relatives à cette campagne sont données dans la première ligne du tableau 1 précédent.
2. On suppose maintenant que la table `Evaluations` contient uniquement les onze enregistrements présentés dans le tableau 1.
   (a) Combien de lignes s'affichent après l'exécution de la requête suivante ?

```
SELECT auteur FROM Evaluations
```

   (b) Recopier les lignes issues de la requête suivante :

```
SELECT Nom_evaluation, Date FROM Evaluations WHERE auteur= "Peltier"
```

   (c) Rédiger une requête permettant de connaître le nom des campagnes prévoyant un entraînement ciblé sur le web (`Code_competenc` de 452).
3. Le système de gestion de bases de données dispose également d'une table `resultats` dont la structure est la suivante :

attribut	type
Code_evaluation	CHAR
Num_eleve	INT
Score	INT

Si l'élève s'est connecté à la campagne mais n'a pas cliqué sur « envoyer les résultats », son score vaut `-1`.

(a) Qu'imposerait le choix du couple `(Code_evaluation, Num_eleve)` comme clé primaire pour la table `resultats` ?

Un extrait de la relation est donné ci-dessous :

Code_evaluation	Num_eleve	Scores
PRJUYR491	17	300
CRLYYR439	654	-1
PRJUYR491	1454	220
RTKVLX656	554	255
DZLYYR479	17	-1
XJVBTX585	1664	12
CRLYYR439	18703	0
PRJUYR491	1565	422
XJVBTX585	12	643
CRLYYR439	168	19
DZLYYR479	17	140
XJVBTX585	1658	647

(b) Ecrire une requête permettant d'obtenir les numéros des élèves `(Num_eleve)` qui ont travaillé la compétence `532`.

4. (a) Proposer la structure d'une table `eleves` permettant d'identifier les noms, prénoms et les classes des élèves.

(b) Proposer une clef primaire pour cette table.

Exercice 4 (POO - 4 points)
Simon souhaite créer en Python le jeu de cartes « la bataille » pour deux joueurs. Les questions qui suivent demandent de reprogrammer quelques fonctions du jeu.

On rappelle ici les règles du jeu de la bataille :

- ⋆ Préparation :
  - Distribuer toutes les cartes aux deux joueurs.
  - Les joueurs ne prennent pas connaissance de leurs cartes et les laissent en tas face cachée devant eux.
- ⋆ Déroulement :
  - A chaque tour, chaque joueur dévoile la carte du haut de son tas.
  - Le joueur qui présente la carte ayant la plus haute valeur emporte les deux cartes qu'il place sous son tas.
  - Les valeurs des cartes sont dans l'ordre de la plus forte à la plus faible : As, Roi, Dame, Valet, 10, 9, 8, 7, 6, 5, 4, 3 et 2 (la plus faible)
- ⋆ Si deux cartes sont de même valeur, il y a « bataille » :
  - Chaque joueur pose alors une carte face cachée, suivie d'une carte face visible sur la carte dévoilée précédemment.
  - On recommence l'opération s'il y a de nouveau une bataille sinon, le joueur ayant la valeur la plus forte emporte tout le tas.
- ⋆ Lorsque l'un des joueurs **possède toutes les cartes du jeu**, la partie s'arrête et ce dernier gagne.

Pour cela Simon crée une classe Python `Carte`. Chaque instance de la classe a deux attributs : un pour sa valeur et un pour sa couleur. Il donne au valet la valeur 11, à la dame la valeur 12, au roi la valeur 13 et à l'as la valeur 14. La couleur est une chaîne de caractères : `"trefle"`, `"carreau"`, `"coeur"` ou `"pique"`.

1. Simon a écrit la classe Python `Carte` suivante, ayant deux attributs `valeur` et `couleur`, et dont le constructeur prend deux arguments : `val` et `coul`.
   (a) Recopier et compléter les `.....` des lignes 3 et 4 ci-dessous.

```
1 class Carte:
2     def __init__(self, val, coul):
3         ..... .valeur = .....
4         ..... = coul
```

   (b) Parmi les propositions ci-dessous, quelle instruction permet de créer l'objet « 7 de coeur » sous le nom `c7` ?
   - `c7.__init__(self, 7, "coeur")`
   - `c7 = Carte(self, 7, "coeur")`
   - `c7 = Carte(7, "coeur")`
   - `from Carte import 7, "coeur"`
2. On souhaite créer le jeu de cartes. Pour cela, on écrit une fonction `initialiser` :
   - sans paramètre ;
   - qui renvoie une liste de 52 objets de la classe `Carte` représentant les 52 cartes du jeu.

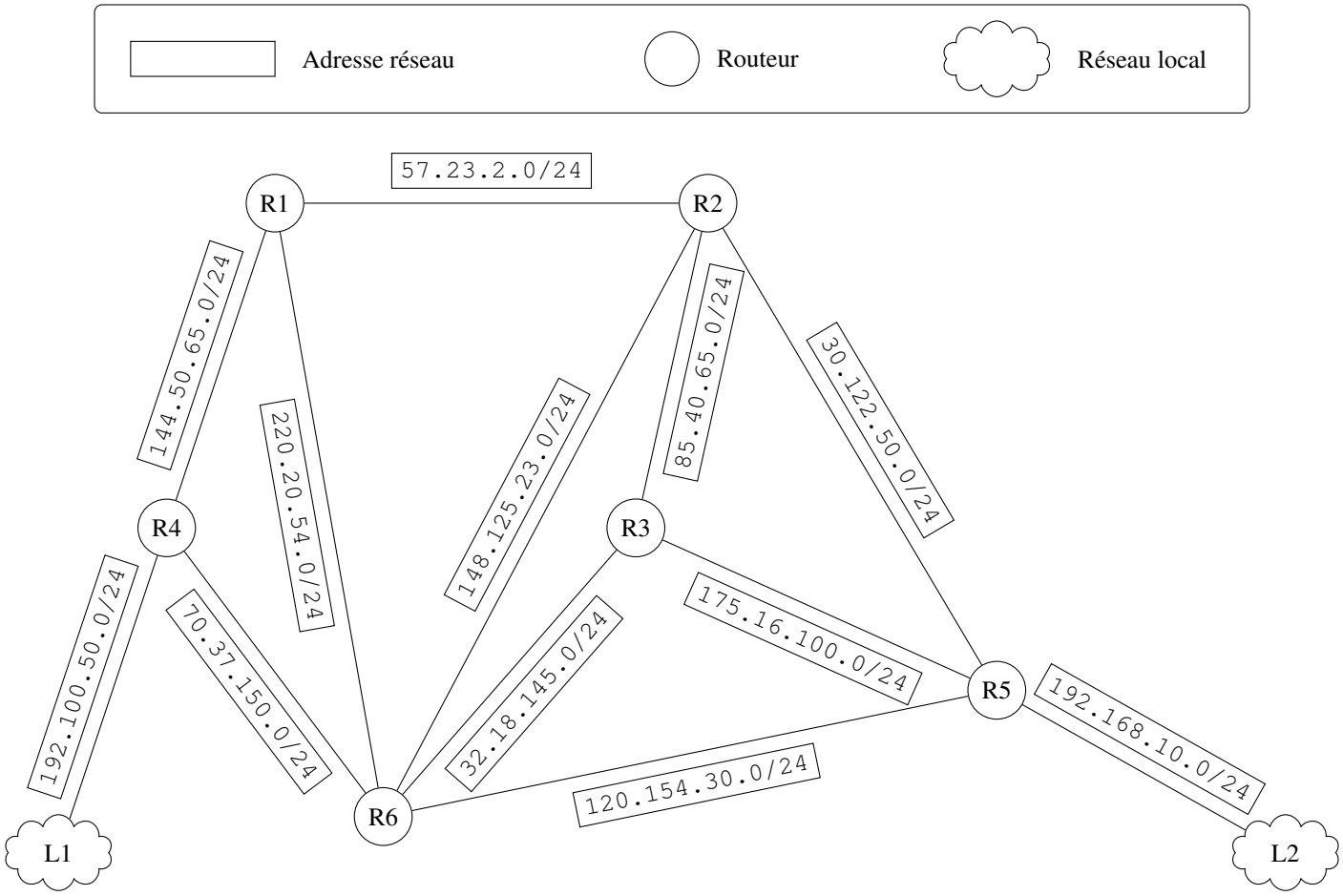
   Voici une proposition de code. Recopier et compléter les lignes suivantes pour que la fonction réponde à la demande :

```
def initialiser() :
    jeu = []
    for c in ["coeur", "carreau", "trefle", "pique"] : # couleur carte
        for v in range(...) : # valeur carte
            carte_cree = ...
            jeu.append(carte_cree)
    return jeu
```

3. On rappelle que dans une partie de bataille, les deux joueurs tirent chacun une carte du dessus de leur tas, et celui qui tire la carte la plus forte remporte les deux cartes et les place en dessous de son tas.
   Parmi les structures linéaires de données suivantes : Tableau, File, Pile, quelle est celle qui modélise le mieux un tas de cartes dans ce jeu de la bataille ? Justifier votre choix.
4. Ecrire une fonction `comparer` qui prend en paramètres deux objets `carte1` et `carte2` de la classe `Carte`. Cette fonction renvoie :
   - `0` si la force des deux cartes est identique ;
   - `1` si la carte `carte1` est strictement plus forte que la carte `carte2` :
   - `-1` si la carte `carte2` est strictement plus forte que la carte `carte1`.

Exercice 5 (Architecture matérielle, OS et réseaux - 4 points)
On représente schématiquement un réseau dans lequel :

- ⋆ L1 et L2 sont des réseaux locaux ;
- ⋆ R1, R2, R3, R4, R5, et R6 sont des routeurs.



1. Dans cet exercice, les adresses IPV4 sont composées de quatre octets `O1.O2.O3.O4`, où `O1`, `O2`, `O3` et `O4` sont les représentations décimales de chacun des octets.
   La notation "`O1.O2.O3.O4 / n `" est appelée la notation CIDR. En notation CIDR, l'adresse IP d'une machine est composée d'une adresse IPv4 et d'une indication sur le masque de sous réseau. Par exemple : `172.16.1.10 / 16` signifie :
   - ⋆ Adresse IP décimale : `172.16.1.10`
   - ⋆ Masque de sous-réseau en notation CIDR : `16`

   La notation CIDR `/16` signifie que le masque de sous-réseau a les 16 bits de poids fort de son adresse IP à la valeur 1. Pour notre exemple, le masque de sous-réseau est : 11111111.11111111.00000000.00000000
   (a) Une machine cliente du réseau local L2 a pour adresse IPv4 : `192.168.10.31 / 24`
   Son adresse IP binaire est : `00110110. 00100101. 01111010.O4 / 24`
   Donner la conversion binaire du quatrième octet `O4` de l'adresse IPv4 de la machine cliente.
   (b) Donner le masque de sous-réseau en notation binaire puis en notation décimale.
   (c) Combien peut-on connecter de machines sur ce sous-réseau ?

Les adresses IP des interfaces des routeurs sont données suivant la stratégie suivante : le dernier octet (`O4`) a pour valeur décimale le numéro du routeur. Exemples :

Réseau `57.23.2.0`		Réseau `148.128.23.0`	
Routeur	IP de l'interface	Routeur	IP de l'interface
R1	`57.23.2.1`	R2	`148.125.23.2`
R2	`57.23.2.2`	R6	`148.125.23.6`

On communique ci-dessous des **extraits** de la table de routage des routeurs R1, R2, R3, R4 et R6 :

Routeur	Réseau destinataire	Passerelle	Interface
R1	`192.168.10.0`	`57.23.2.2`	`57.23.2.1`
R2	`192.168.10.0`	`30.122.50.5`	`30.122.50.2`
R3	`192.168.10.0`	`175.16.100.5`	`175.16.100.3`
R4	`192.168.10.0`	`70.37.150.6`	`70.37.150.4`
R6	`192.168.10.0`	`120.154.30.5`	`120.154.30.6`

2. Un paquet de données part du réseau local L1 pour aller vers L2.
   (a) En utilisant le schéma du réseau et l'extrait de la table de routage du routeur R4, vers quel routeur R4 envoie-t-il ce paquet, R1 ou R6 ? Justifier
   (b) Nommer les routeurs traversés par ce paquet lorsqu'il va de L1 à L2.
3. La liaison est coupée entre R4 et R6 :
   (a) Sachant que ce réseau utilise le protocole RIP (distance minimale en nombre de sauts), donner l'un des deux chemins possibles que pourra suivre un paquet de données allant de L1 vers L2.
   (b) Dans les extraits de la table de routage, pour le chemin de la question 3.a., quelle(s) ligne(s) sera ou seront modifiée(s).
4. La liaison entre R4 et R6 est rétablie. Par ailleurs, on décide d'utiliser le protocole OSPF (distance liée au coût C minimal des liaisons) pour effectuer le routage. Le coût C des liaisons entre les routeurs est conditionné par la bande passante (BP) des liaisons entre les routeurs. Le coût C est donné par la formule :

$$C = \frac{10^8}{\text{BP}}.$$

   La bande passante (BP) peut s'exprimer en Mégabits par seconde. Plus BP est importante, plus le coût C des liaisons diminue. Le coût des liaisons est donné dans le tableau ci-dessous :

Liaison	R1 – R4	R1 – R2	R1 – R6	R2 – R3	R2 – R5	R2 – R6	R3 – R5	R3 – R6	R4 – R6	R5 – R6
Coût	100	1	1	1	100	10	1	10	100	100

5. (a) Dessiner le réseau en y ajoutant les coûts entre les connexions. Déterminer le chemin parcouru par un paquet partant du réseau L1 et arrivant au réseau L2 en utilisant le protocole OSPF (le moindre coût).
   (b) Indiquer pour quel(s) routeur(s) l'extrait de la table de routage sera modifié pour un paquet à destination de L2, avec le protocole OSPF.

Annexe 1 – Langage SQL

(à ne pas rendre avec la copie)

⋆ **Types de données :**

`CHAR`	Chaîne de caractères
`INT`	Nombre entier de -2^{31} à $2^{31}-1$ (signé) ou de 0 à $2^{32}-1$ (non signé)
`FLOAT`	Réel à virgule flottante
`DATE`	Date format `AAAA-MM-JJ`
`DATETIME`	Date et heure format `AAAA-MM-JJ-HH:MM:SS`

⋆ **Quelques exemples de syntaxe SQL :**

- Insérer des enregistrements :
```
INSERT INTO Table (attribut1, attribut2) VALUES(valeur1 , valeur2)
```

- Modifier des enregistrements :
```
UPDATE Table SET attribut1=valeur1, attribut2=valeur2 WHERE Selecteur
```

- Supprimer des enregistrements :
```
DELETE FROM Table WHERE Selecteur
```

- Sélectionner des enregistrements :
```
SELECT attributs FROM Table WHERE Selecteur
```

- Effectuer une jointure :
```
SELECT attributs FROM TableA JOIN TableB ON TableA.cle1=TableB.cle2 WHERE Selecteur
```