Université de Carthage Institut Préparatoire aux Etudes d'Ingénieur de Nabeul Département : Mathématiques	IPEIN المعهد التحضيري للدراسات الهندسية بنابل Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul	Année universitaire : 2019/2020 Filière : SM/SP/ST Niveau d'étude : 2ème année Semestre : 2 Nombre de pages : 7 pages Date : 19/06/2020 Durée : 2h00

Concours blancs : Epreuve d'Informatique

Exercice 1 : Les bases de données (10 pts)

Dans les réseaux sociaux, on peut créer des groupes, et d'inviter des personnes à ces groupes. Dans le but de faire des statistiques sur les groupes et leurs membres, nous avons créé une base de données relationnelle composée de trois tables : **Personnes**, **Membres** et **Groupes**.

A- Structure de la table 'Personnes' :

La table '**Personnes'** contient les personnes qui utilisent les réseaux sociaux. Ces personnes peuvent créer des groupes et aussi peuvent se joindre aux groupes, et devenir membres de ces groupes.

- Le champ **code** de type entier, contient un entier positif unique pour chaque personne ;
- Le champ **nom** de type texte, contient le nom de chaque personne ;
- Le champ **prenom** de type texte, contient le prénom de chaque personne ;
- Le champ **dateNaiss** de type date, contient la date de naissance de chaque personne.

Exemples d'enregistrements dans la table **'Personnes'** :

code	nom	prenom	dateNaiss
1	Boudara	Mohamed	1996-10-15
2	Mernissi	Salima	2000-03-29
3	Sabri	Ahmed	2001-02-20
…	…	…	…

B- Structure de la table 'Groupes' :

La table '**Groupes'** contient des informations relatives à chaque groupe :

- Le champ **id** de type entier, contient les identifiants des groupes.
  L'identifiant de chaque groupe est unique ;
- Le champ **nom** de type texte, contient les noms des groupes ;
- Le champ **description** de type texte, contient un texte qui précise les objectifs de chaque groupe ;
- Le champ **codeCr**, de type entier, contient les codes des personnes ayant créé les groupes ;
- Le champ **dateCr**, de type date, contient la date de création de chaque groupe.

Exemples d'enregistrements dans la table '**Groupes'** :

id	nom	description	codeCr	dateCr
253	Amis	Les amis les plus proches	2	2017-07-09
471	Fans	Discussion à propos de divers sujets	6	2016-09-14
96	Famille	Les membres de la grande famille	25	2016-10-10

…	…	…	…	…

C- Structure de la table ‘Membres’ :

La table ‘**Membres’** est composée de trois champs :

- Le champ **code** de type entier, contient les codes des personnes qui ont accepté l’invitation pour devenir membres aux groupes ;
- Le champ **id** de type entier, contient les identifiants des groupes ;
- Le champ **dateM** de type date, contient les dates auxquelles chaque personne a rejoint un groupe.

Exemples d’enregistrements dans la table ‘**Membres’** :

code	id	dateM
1	253	2017-09-18
3	471	2018-01-30
4	253	2016-09-25
5	253	2018-01-22
7	253	2017-07-29
8	471	2016-11-10
…	…	…

1.5 pts **Travail à faire :**

1. Déterminer les clés primaires et les clés étrangères dans les tables de cette base de données.

   Justifier votre réponse.

1 pt Écrire, en langage SQL, les requêtes qui donnent les résultats suivants :

2. Les personnes qui sont nées le mois 6, triées dans l’ordre croissant des âges (du moins âgée au plus âgée).

1 pt

3. Les identifiants des groupes, les dates de création des groupes et les noms et prénoms des personnes ayant créé ces groupes.

1 pt

4. Écrire la requête de la question **3** en algèbre relationnelle.

1 pt

5. Les noms et les descriptions des groupes qui contiennent au moins 1000 personnes, triés dans l’ordre décroissant des nombres de personnes.

1.5 pts

6. Les codes, les noms et les prénoms de toutes les personnes (créateur et membres) qui appartiennent au groupe ayant le nom : 'Sport'.

1.5 pts

1.5 pts

7. Supprimer les personnes qui n’appartiennent à aucun groupe.

8. Modifier dans la table 'Membres', pour que tous les membres du groupe ayant le nom ‘Sport’ deviennent membres du groupe ayant le nom ‘Voyage’.

Problème : Cryptographie (10 pts)

Le but de ce problème est d’implémenter une méthode de chiffrement et de déchiffrement.

PARTIE 1 : CHIFFREMENT

Description

Soit un message de longueur **n** (avec n pair), écrit en lettres majuscules de A à Z, représenté par une chaine de caractères de taille **n**.

L’idée du chiffrement est de grouper les lettres du message par bloc de 2, puis de les chiffrer bloc par bloc en utilisant la méthode suivante :

Les lettres sont codées par leur rang dans l’alphabet comme le montre le tableau suivant :

Lettre	A	B	C	D	E	F	G	H	I	…	S	T	U	V	W	X	Y	Z
Code	0	1	2	3	4	5	6	7	8	…	18	19	20	21	22	23	24	25

Les codes P_k et P_{k+1} de deux lettre successives d’un texte seront chiffrées C_k et C_{k+1} en utilisant la formule :

$$\begin{pmatrix} C_k \\ C_{k+1} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} P_k \\ P_{k+1} \end{pmatrix} (mod\ 26)$$

Avec :

- P_k et P_{k+1} respectivement les codes des lettres k et $k+1$ du texte clair (texte original non chiffré).
- C_k et C_{k+1} respectivement les codes des lettres k et $k+1$ du texte chiffré.
- La matrice $Mc = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ du chiffrement vérifiant les caractéristiques suivantes :
    - Les éléments a, b, c et d du Mc doivent être des **entiers positifs**.
    - La matrice Mc doit être **inversible.** La matrice inverse existe si $(ad-bc)$ est impair et n’est pas multiple de 13.

Exemple de chiffrement :

On souhaite chiffrer le mot « ELECTION »

Mot à chiffrer	E	L	E	C	T	I	O	N

Avec la matrice de chiffrement $Mc = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$

D’après le tableau de codage, les codes correspondants aux lettres du mot « ELECTION » sont :

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7
4	11	4	2	19	8	14	13

Les deux premières lettres seront donc chiffrées ainsi :

$$\begin{pmatrix} C_0 \\ C_1 \end{pmatrix} = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 4 \\ 11 \end{pmatrix} (mod\ 26) = \begin{pmatrix} 80 \\ 97 \end{pmatrix} (mod\ 26) = \begin{pmatrix} 2 \\ 19 \end{pmatrix}$$

Les nombres 2 et 19 correspondent respectivement aux lettres C et T.

En procédant de même avec les paires des lettres suivantes, on obtient les codes chiffrés suivants :

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
	2	19	18	8	21	21	22	5
Mot chiffré	C	T	S	I	V	V	W	F

1 pt

Travail demandé :

1 pt On suppose avoir défini une liste **Alphabet** contenant les lettres majuscules de A à Z suivante :

alphabet = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', … , 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

1 pt

1. Ecrire une fonction Python **taille** permettant de saisir et retourner un entier pair strictement positif et inférieur ou égal à un entier *NMAX* donné passé en paramètres.

1 pt

2. Ecrire une fonction Python **code_car**, qui à partir d'un paramètre **c** de type **str**, retourne l'indice du caractère **c** dans l'**alphabet** si **c** appartient à l'**alphabet** et **-1** sinon.

2 pts

3. Ecrire une fonction Python **saisie_msg** permettant de saisir une chaine de caractères de **n** lettres, en vérifiant l'appartenance à la liste **Alphabet** de chacune des lettres saisies.
4. Ecrire une fonction Python **saisie_matc** permettant de saisir la matrice de chiffrement **Mc**.
5. Ecrire une fonction Python **chiffrer** permettant de chiffrer dans une chaine de caractères **msg_chiffre**, un message clair donné représenté par une chaine de caractères **msg** de longueur **n**.

PARTIE 2 : DECHIFFREMENT

Pour déchiffrer un message, on multiplie les codes des lettres du message chiffré deux par deux par la matrice de déchiffrement.

$$\begin{pmatrix} P_k \\ P_{k+1} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} \begin{pmatrix} C_k \\ C_{k+1} \end{pmatrix} (mod\ 26)$$

Avec :

- C_k et C_{k+1} respectivement les codes des lettres k et $k+1$ du texte chiffré.
- P_k et P_{k+1} respectivement les codes des lettres k et $k+1$ du texte clair.

- La matrice $Md=\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1}$ est la matrice de déchiffrement. Le calcul de *Md* est donné par :

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = m\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} mod\, 26$$

Où :

- $m \in \{1,3,5,7,9,11,15,17,19,21,23,25\}$
- *m* respecte la condition $((ad-bc)*m)\, mod\, 26 = 1$

Exemple de déchiffrement :

Considérant la matrice de chiffrement $\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$

La matrice de déchiffrement correspondante est : $Md=\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}^{-1} = m\begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} mod\, 26$

La première valeur de m tel que $m \in \{1,3,5,7,9,11,15,17,19,21,23,25\}$ vérifiant la condition $((ad-bc)*m)\, mod\, 26 = 1$ est celle retenue. Dans ce cas cette valeur est égale à 23.

Ce qui donne : $Md=\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = 23\begin{pmatrix} 7 & -4 \\ -5 & 9 \end{pmatrix} mod\, 26 = \begin{pmatrix} 5 & 12 \\ 15 & 25 \end{pmatrix}$

On se propose de déchiffrer le mot « **CTSIVVWF** » représenté par le tableau suivant :

Mot à déchiffrer	C	T	S	I	V	V	W	F

D'après le tableau de codage, les codes C_k correspondants au mot « **CTSIVVWF** » sont :

C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
2	19	18	8	21	21	22	5

Les deux premières lettres du mot chiffré seront donc ainsi :

$$\begin{pmatrix} P_0 \\ P_1 \end{pmatrix} = \begin{pmatrix} 5 & 12 \\ 15 & 25 \end{pmatrix} \begin{pmatrix} 2 \\ 19 \end{pmatrix} (mod\, 26) = \begin{pmatrix} 4 \\ 11 \end{pmatrix}$$

Les nombres 4 et 11 correspondent respectivement aux lettres E et L.

[1 pt] En procédant de même avec les paires des lettres suivantes, on obtiendra les codes chiffrés P_k suivants :

	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7
	4	11	4	2	19	8	14	13
Mot déchiffré	E	L	E	C	T	I	O	N

Travail demandé :

6. Ecrire une fonction Python **init_liste**, qui remplit une liste **E** avec des entiers impairs de **1** à **26** et non multiple de de **13**.
7. Ecrire une fonction Python **cree_Md** qui crée, à partir de la matrice **Mc** de chiffrement, la matrice **Md** de déchiffrement en utilisant le principe de calcul décrit précédemment.
8. Ecrire une fonction Python **dechiffrer** permettant de déchiffrer dans une chaine de caractère **msgDc**, un message chiffré donné représenté par une chaine de caractère **msgC** de longueur n.

1 pt

2 pts