RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION	EXAMEN DU BACCALAURÉAT	Session de contrôle	2024
	Épreuve : **Algorithmique et Programmation**	Section : **Sciences de l'informatique**	
	Durée : **3h**	Coefficient de l'épreuve : **2**	

N° d'inscription | | | | | | |

Le sujet comporte 4 pages numérotées de 1 sur 4 à 4 sur 4.

Exercice 1 : (3 points)

Soit l'algorithme ci-dessous de la fonction "**Inconnue**" avec **ch1** et **ch2** deux chaines, non vides, formées uniquement par des lettres minuscules distinctes :

```
Fonction Inconnue(ch1, ch2 : Chaîne de caractères) : Chaîne de caractères
DEBUT
    Si ch1 = "" Alors
            Retourner ""
    Sinon Si Pos (ch1[0], ch2)  = -1 Alors
            Retourner  Inconnue(Sous_Chaine (ch1, 1, Long (ch1)), ch2)
    Sinon
            Retourner  ch1[0] + Inconnue(Sous_Chaine (ch1, 1, Long (ch1)), ch2)
    FinSi
FIN
```

Travail demandé :

1) Donner une trace d'exécution manuelle vérifiant le résultat obtenu pour chacun des deux cas suivants :
   - **Cas1 :** Pour ch1 = "**bac**" et ch2 = "**courage**", **Inconnue("bac","courage")** = "**ac**".
   - **Cas2 :** Pour ch1 = "**image**" et ch2 = "**matrice**", **Inconnue("image","matrice")** = "**imae**"
2) En déduire le rôle de la fonction **Inconnue**.
3) Ecrire un algorithme de la fonction **Inconnue** en utilisant un traitement itératif donnant le même résultat.

Exercice 2 : (3 points)

Pour obtenir une valeur approchée de $ln(2)$, avec ln est le logarithme népérien, on calcule la somme des **n** premiers termes de la suite suivante :

$$ln(2) = 1 - \frac{1}{2} + \frac{1}{3} - \cdots + \frac{(-1)^{n-1}}{n}$$

Travail demandé :

Ecrire un algorithme d'une fonction **CalculLn2(epsilon)** qui retourne une valeur approchée de ***ln(2)***. Le calcul s'arrête lorsque la valeur absolue du dernier terme ajouté est inférieure ou égale à **epsilon**.

Exercice 3 : (6 points)

Pour deux entiers naturels **n** et **k**, le coefficient binomial qu'on note **Cb (n,k)** est défini comme étant le nombre de sous-ensembles différents, à **k** éléments, que l'on peut former à partir d'un ensemble contenant **n** éléments.

Le coefficient binomial **Cb (n,k)** est un entier naturel vérifiant l'un des trois cas suivants :

Cas 1 : Cb (n,k) = 0 si n < k

Cas 2 : Cb (n,k) = 1 si k = 0 ou k = n

Cas 3 : Cb (n,k) = Cb (n-1,k-1) + Cb (n-1,k) si $1 \leq k \leq n-1$

Exemples :

- Cb (12,315) = 0 car 12 < 315 : le cas appliqué est **Cas 1**
- Cb (25,0) = 1 car k = 0 : le cas appliqué est **Cas 2**
- Cb (2,2) = 1 car k = n : le cas appliqué est **Cas 2**
- Cb (3,2) = Cb (2,1) + Cb (2,2) : Le cas appliqué est **Cas 3**
  = Cb (1,0) + Cb (1,1) + Cb (2,2) : Le cas appliqué est **Cas 3**
  = 1 + 1 + 1 : Le cas appliqué est **Cas 2**
  = 3

On se propose de remplir un tableau d'enregistrements **T** à partir d'un fichier texte nommé "**Coeff.txt**" qui contient dans **chaque ligne un couple** d'entiers ayant comme format **(n,k)**. Chaque enregistrement du tableau **T** contiendra les trois champs suivants :

- **n :** La première valeur du couple binomial de type entier.
- **k :** La deuxième valeur du couple binomial de type entier.
- **r :** Le résultat du coefficient binomial de type entier.

Exemple :

Pour le contenu du fichier "**Coeff.txt**" suivant :

(12,315)
(3,2)
(2,2)
(25,0)
(4,3)

Le contenu du tableau **T** sera :

0			1			2			3			4		
12	315	0	3	2	3	2	2	1	25	0	1	4	3	4

Travail demandé :

1) Sachant que le fichier "**Coeff.txt**" contient au maximum **50** couples d'entiers, déclarer un type pour le tableau **T** ainsi que tout type nécessaire à sa déclaration.
2) Ecrire un algorithme d'une procédure nommée **Remplir (Ch , T)** qui permet de remplir le tableau **T** comme décrit précédemment.

   NB :
   - Le candidat n'est pas appelé à remplir le fichier "**Coeff.txt**". On rappelle que chaque ligne du fichier contient un couple d'entiers ayant le format **(n,k)**.
   - Le paramètre **Ch** utilisé dans l'entête du module **Remplir** est une chaine de caractères contenant l'emplacement et le nom physique du fichier "**Coeff.txt**".
   - La procédure **Remplir** doit prendre en charge l'ouverture et la fermeture du fichier "**Coeff.txt**".

Exercice 4 : (8 points)

On se propose de dissimuler une image **Img2** dans une autre image **Img1** de manière discrète en utilisant les étapes de dissimulation détaillées ci-après.

Les deux images sont représentées par deux matrices de même dimension **L*C** pixels. Un pixel est un enregistrement composé de trois champs **R**, **V** et **B** de type entier (code décimal entre **0** et **255**) représentant respectivement les couleurs Rouge, Verte et Bleue.

Exemple : l'enregistrement | 80 | 120 | 43 | représente un pixel dont le champ **R = 80**, le champ **V = 120** et le champ **B = 43**.

Etapes de dissimulation :

- Pour chaque deux pixels de même position dans les images **Img1** et **Img2** :
  - **Conversion binaire :** convertir en binaire les deux codes décimaux de la couleur rouge des deux pixels pour obtenir ainsi deux nombres binaires chacun de **8** chiffres : **bin1** pour le pixel de **Img1** et **bin2** pour le pixel de **Img2**.
  - **Dissimulation :** remplacer les **4** derniers bits (les plus à droite) de **bin1** par les **4** premiers bits (les plus à gauche) du nombre binaire **bin2**.
  - **Conversion décimale :** convertir la valeur binaire résultat de l'étape "Dissimulation" en décimale qui devient alors la nouvelle valeur de la couleur rouge du pixel de l'image **Img1**.
- Refaire les trois étapes précédentes pour la couleur verte puis pour la couleur bleue.

Exemple :

Pour les deux matrices suivantes **Img1** et **Img2** de taille **2*2** :

Img2

	0	1
0	80 120 43	121 0 16
1	120 15 12	155 200 190

Img1

	0	1
0	190 0 10	180 20 30
1	160 20 120	155 60 90

La matrice **Img1**, après dissimulation, devient :

Img1

	0	1
0	181 7 2	180 16 17
1	167 16 112	153 60 91

En effet :

Pour le 1er pixel de Img2 (**Img2[0,0]**) et le 1er pixel de Img1 (**Img1[0,0]**), les étapes de dissimulation donne :

		R	V	B
1er pixel de Img2		80	120	43
1er pixel de Img1		190	0	10
Conversion binaire	bin2	$(\underline{0101}0000)_2$	$(\underline{0111}1000)_2$	$(\underline{0010}1011)_2$
	bin1	$(1011\underline{1110})_2$	$(0000\underline{0000})_2$	$(0000\underline{1010})_2$
Dissimulation		$(1011\underline{0101})_2$	$(0000\underline{0111})_2$	$(0000\underline{0010})_2$
Conversion décimale		181	7	2

Ainsi **Img1[0,0]** devient

181	7	2

Travail demandé :

1) Déclarer un type pour les matrices représentant les deux images **Img1** et **Img2** sachant que $2 \leq L \leq 400$ et $2 \leq C \leq 500$, ainsi que tous les types nécessaires à sa déclaration.
2) Ecrire un algorithme d'un module **Resultat (Img1 , Img2 , L , C)** permettant de dissimuler une image représentée par une matrice **Img2** dans une image représentée par une matrice **Img1**, en appliquant le principe décrit précédemment sachant que les deux matrices **Img1** et **Img2** ont le même nombre de lignes **L** et le même nombre de colonnes **C**.
3) Ecrire un algorithme d'un module nommé **Image (Img1 , L , C , Ch)** permettant de générer un fichier texte **F** contenant les représentations décimales des couleurs des pixels de l'image résultat **Img1**. Une ligne du fichier **F** correspond à une ligne de la matrice résultat et contient la concaténation des codes des couleurs des pixels structurés comme suit :
   - les codes des couleurs d'un pixel sont séparés par un tiret "-".
   - les pixels sont séparés par le caractère "#".

NB :

- Le paramètre **Ch** utilisé dans l'entête du module **Image** est une chaine de caractères contenant l'emplacement et le nom physique du fichier **F**.
- Le module demandé doit prendre en charge la création et la fermeture du fichier **F**.

Exemple :

Pour la matrice résultat **Img1** précédente, la concaténation des pixels donne pour :

- la 1ère ligne : **181-7-2#180-16-17**
- la 2ème ligne : **167-16-112#153-60-91**

Et ainsi le contenu du fichier **F** est :

F

181-7-2#180-16-17
167-16-112#153-60-91