

RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION ***** EXAMEN DU BACCALAURÉAT	Épreuve : <b>ALGORITHMIQUE ET PROGRAMMATION</b>
	Section : <b>Sciences de l'informatique</b>
	Durée : <b>3H</b> Coefficient : <b>2.25</b>
<b>SESSION 2016</b>	<b>Session de contrôle</b>

Section : ..... N° d'inscription : ..... Série : .....

Nom et prénom : .....

Date et lieu de naissance : .....

Signatures des surveillants  
 .....  
 .....



*Le sujet comporte 3 pages numérotées de 1/3 à 3/3.  
 Les réponses à la question 1 de l'exercice 1 doivent être rédigées sur la page 1/3  
 qui doit être remise avec la copie.*

**Exercice 1 : (3 points)**

Soit une fonction **Symetrie** qui vérifie si le contenu d'un fichier d'entiers **F**, déjà rempli, est symétrique. On propose ci-dessous un algorithme de cette fonction, contenant trois erreurs dans le choix des fonctions et des procédures prédéfinies utilisées :

```

0) DEF FN Symetrie (Var F : fiche_ent) : booléen
1) Recréer(F) ; S ← Vrai ; t ← taille_fichier(F)
   Pour i de 1 à fin_fichier(F) div 2 Faire
     Pointer(F,i)
     Lire_nl(F,M)
     Pointer(F,t - i + 1)
     Lire(F,N)
     S ← (S et (M =N))
   Fin Pour
   Fermer(F)
2) Symetrie ← S
3) Fin Symetrie
  
```

**Travail à faire :**

1) Compléter le tableau ci-dessous en remplissant la première colonne par les fonctions ou les procédures prédéfinies dont l'utilisation est erronée et la deuxième colonne par les fonctions ou les procédures prédéfinies adéquates :

Fonction ou procédure prédéfinie dont l'utilisation est erronée	Fonction ou procédure prédéfinie adéquate

RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION ***** EXAMEN DU BACCALAURÉAT	Épreuve : <b>ALGORITHMIQUE ET PROGRAMMATION</b>	
	Section : <b>Sciences de l'informatique</b>	
	Durée : <b>3H</b>	Coefficient : <b>2.25</b>
<b>SESSION 2016</b>		<b>Session de contrôle</b>

- 2) Dans le but d'améliorer l'algorithme proposé, réécrire la séquence n°1 en utilisant la structure itérative adéquate et en tenant compte des corrections apportées dans la question n°1.

### Exercice 2 : (4,75 points)

Soit la suite de Perrin définie par :

$$\begin{cases} U_0 = 3, U_1 = 0, U_2 = 2 \\ U_n = U_{n-2} + U_{n-3} \text{ pour } n \geq 3 \end{cases}$$

Cette suite vérifie la propriété suivante : "**Pour tout entier  $n > 2$ , si  $n$  divise  $U_n$  alors  $n$  est un nombre premier**".

**Travail à faire :**

- 1- Ecrire une analyse d'un module intitulé "**Verif\_pr**", permettant de vérifier si un entier **N** est premier et ce, en utilisant la propriété décrite précédemment.

**NB :** **N** est déjà saisi dans le programme appelant.

- 2- Ecrire un algorithme d'un module récursif intitulé "**Verif\_geo**", permettant de vérifier si la suite **U** est géométrique sur un intervalle  $[a, b]$  donné.

**NB :**

- Une suite **U** est dite géométrique s'il existe un réel **q** (appelé raison) tel que pour tout entier **n** de l'intervalle  $[a, b]$ ,  $U_{n+1} = q \times U_n$ .
- **a** et **b** sont déjà saisis au niveau du programme appelant (avec  $b \geq a + 2$ ).

### Exercice 3 : (3,5 points)

Un **nombre primaire**, également appelé puissance première, est une puissance à exposant entier positif non nul d'un nombre premier.

**Exemples :**

- 5, 9 et 16 sont des nombres primaires, car  $5 = 5^1$ ,  $9 = 3^2$  et  $16 = 2^4$ .
- 6 et 36 ne sont pas des nombres primaires, car on ne peut pas les écrire sous forme d'une puissance à exposant entier positif non nul d'un nombre premier.

**Travail à faire :**

Ecrire un algorithme d'un module intitulé "**Nb\_primaire**" qui affiche les **N** premiers nombres primaires.

**NB :** **N** est déjà saisi au niveau du programme appelant.

### Exercice 4 : (3,5 points)

Soit un tableau **T\_DN** contenant les dates de naissance de **N** personnes. On se propose de trier le tableau **T\_DN** par ordre croissant.

**Travail à faire :**

- 1- Donner une structure de données adéquate pour représenter une date de naissance.
- 2- En utilisant la structure de donnée proposée dans la question n°1, écrire un algorithme d'un module intitulé "Tri" qui permet de trier les N éléments du tableau T\_DN par ordre croissant de la date de naissance.

**NB:** N est déjà saisi au niveau du programme appelant.

**Exercice 5 : (5,25 points)**

Soit M une matrice carrée d'ordre N (avec  $N \leq 15$ ) remplie par des lettres majuscules. On se propose de créer, sur la racine du disque C, un fichier "Symetrie.txt" formé par les lignes et les colonnes symétriques se trouvant dans la matrice M ainsi que leurs nombres. Pour cela, on procède comme suit :

- La première ligne du fichier contient les contenus des lignes symétriques de la matrice M, séparés par le caractère "\*".
- La deuxième ligne du fichier contient le nombre de lignes symétriques contenues dans la matrice M.
- La troisième ligne du fichier contient les contenus des colonnes symétriques de la matrice M, séparés par le caractère "\*".
- La quatrième ligne du fichier contient le nombre de colonnes symétriques contenues dans la matrice M.

**NB :** Une ligne ou une colonne d'une matrice est dite symétrique si la concaténation des caractères contenus dans ses cases forme une chaîne palindrome.

**Exemple :**

Pour N = 5 et la matrice M suivante :

G	B	E	B	G
N	A	R	O	U
E	I	M	L	C
M	A	L	A	M
O	B	E	W	G

- La ligne "MALAM" comme indiquée ci-contre est un exemple de lignes symétriques de la matrice M.
- La colonne "BAIAB" comme indiquée ci-contre est un exemple de colonnes symétriques de la matrice M.

Le fichier "Symetrie.txt" aura le contenu suivant :

```
GBEBG*MALAM
2
BAIAB
1
```

**Travail à faire :**

Ecrire un algorithme d'un module intitulé "L\_C\_Sym" permettant de remplir le fichier "Symetrie.txt" comme décrit précédemment.

**NB:** M et N sont déjà saisis au niveau du programme appelant.