


RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION EXAMEN DU BACCALAURÉAT SESSION 2019	Session principale	
	Épreuve : Algorithmique et Programmation	Section : Sciences de l'informatique
	 Durée : 3h	Coefficient de l'épreuve : 2.25

☞ ☞ ☞ ☞ ☞ ☞

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

Important :

Chaque solution développée par le candidat, sous forme d'une analyse ou d'un algorithme doit être accompagnée d'un tableau de déclarations des objets ayant la forme suivante :

Objet	Type / Nature	Rôle

Exercice 1 (2 points)

Soit l'algorithme suivant de la fonction récursive intitulée **Quoi** :

```

0) DEF FN Quoi (a, b : Réel) : .....
1) Si (a - b) ≥ 0 Alors
    Quoi ← a
    Sinon Quoi ← FN Quoi (b, a)
    Fin Si
2) Fin Quoi

```

Travail demandé :

Reproduire le tableau suivant, puis en se référant à l'algorithme de la fonction **Quoi** et pour chacune des propositions ci-après, remplir la case correspondante par la lettre de la réponse correcte.

Proposition	1	2	3	4
Réponse				

- Le type de la fonction **Quoi** peut être :
 - Octet
 - Réel
 - Entier long
- La condition d'arrêt du traitement récursif est :
 - $(a - b) \geq 0$
 - $Quoi \leftarrow a$
 - $Quoi \leftarrow FN\ Quoi(b, a)$
- Pour $a = 9$ et $b = 12$, le résultat retourné par la fonction **Quoi** est égal à :
 - 9
 - 12
 - 3
- Le rôle de la fonction **Quoi** est de :
 - calculer le PPCM de a et b
 - calculer le PGCD de a et b
 - rechercher le maximum de a et b

Exercice 2 (2,75 points)

La figure ci-dessous représente la courbe de la fonction f définie par $f(x) = \frac{1}{x}$ sur l'intervalle $]0, +\infty[$.

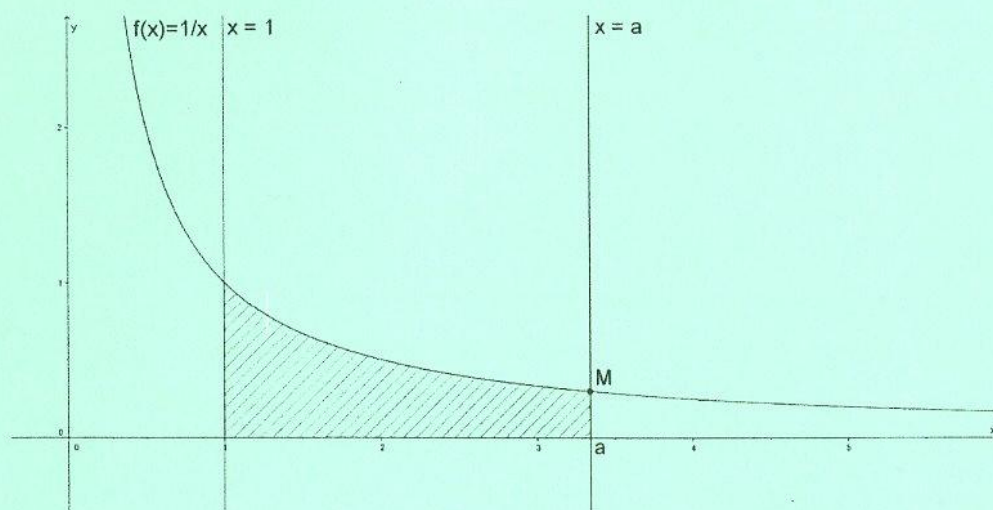


Figure 1

Etant donné que $\int_1^a \frac{1}{x} dx = \begin{cases} < 1 & \text{si } a < e \\ = 1 & \text{si } a = e \\ > 1 & \text{si } a > e \end{cases}$, on remarque que la surface délimitée par les deux droites d'équations $x = 1$ et $x = a$, l'axe des abscisses et la courbe $f(x)$, varie selon la valeur de l'abscisse a du point M (la surface hachurée dans la **Figure 1**). Cette surface sera égale à 1 lorsque la valeur de a est égale au nombre d'Euler e .

Travail demandé :

- Ci-dessous une partie d'un algorithme de la fonction **Surface**, qui permet de calculer la surface hachurée en fonction de l'abscisse a du point M et en utilisant la méthode des rectangles à gauche.

0) DEF FN Surface (a : réel ; n : entier) : réel

1) $S \leftarrow 0$

.....
 $h \leftarrow (a-1) / n$
 Pour i de 1 à n Faire

 $x \leftarrow x + h$
 Fin Pour

2)

3) Fin Surface

T.D.O		
Objet	Type	Rôle
S	Réel	Calculer la somme des $f(x)$
x	Réel	Contenir les valeurs des abscisses
h	Réel	Contenir la largeur des rectangles
i	Entier	Compteur

NB : n représente le nombre de rectangles.

Réécrire l'algorithme de la fonction **Surface** en complétant les vides par les trois instructions convenables à partir de la liste d'instructions suivante :

$x \leftarrow 1$	$S \leftarrow S + 1/x$	$\text{Surface} \leftarrow S * h$
$x \leftarrow 0$	$S \leftarrow S + 1/2(1/x + 1/(x+h))$	$\text{Surface} \leftarrow S * n * h$

- En faisant appel à la fonction **Surface**, écrire un algorithme d'une fonction **Calcul** (a, n) permettant de déterminer une valeur approchée du nombre d'Euler e , qui correspond à une valeur de la surface proche de 1 avec une précision de 10^{-4} .

NB : On pourra calculer le nombre d'Euler e en variant l'abscisse a par pas de 10^{-4} .

Exercice 3 (5,25 points)

Une **fraction** de la forme $\frac{a}{b}$ est dite **irréductible** lorsqu'on ne peut plus la simplifier.

Mathématiquement, une fraction $\frac{a}{b}$ est irréductible si le **PGCD** (a, b) = 1.

Ainsi, pour rendre une fraction $\frac{a}{b}$ irréductible, on divise le numérateur et le dénominateur de cette fraction par le **PGCD** (a, b).

Soit "**Fraction.dat**" un fichier d'enregistrements contenant des fractions représentée chacune par les deux champs suivants :

- **Num** : un entier représentant le numérateur de la fraction.
- **Denom** : un entier représentant le dénominateur de la fraction.

On se propose de créer puis d'afficher, un fichier d'enregistrements intitulé "**Irreduct.dat**" qui devra contenir pour chaque fraction du fichier "**Fraction.dat**" la fraction irréductible correspondante.

Travail demandé :

1. Ecrire un algorithme d'un module permettant de remplir puis d'afficher le fichier "**Irreduct.dat**" comme expliqué ci-dessus.

NB :

- Le candidat n'est pas appelé à remplir le fichier "**Fraction.dat**".
- Les fichiers "**Fraction.dat**" et "**Irreduct.dat**" ont la même structure.
- Il est possible d'utiliser la méthode de la **division euclidienne** pour calculer le **PGCD** de deux entiers **a** et **b** dont le principe se présente comme suit :
 - diviser **a** par **b** pour obtenir un reste **r**,
 - si **r** = 0, le **PGCD** est égal à **b**,
 - si **r** ≠ 0, refaire la division en remplaçant **a** par **b** et **b** par **r** jusqu'à obtenir **r** = 0. Dans ce cas le **PGCD** est égal au dernier reste non nul.

2. Donner une déclaration pour chaque nouveau type utilisé dans la réponse à la question 1.

Problème (10 points)

Parmi les méthodes de chiffrement utilisant un mot-clé, on cite celle décrite ci-après qui permet de crypter un message **msg** ne dépassant pas 18 caractères et formé uniquement de lettres minuscules, de chiffres et d'espaces :

Etape1 : Remplir aléatoirement une matrice carrée **M1** de dimension **6x6** par toutes les lettres alphabétiques minuscules ainsi que tous les chiffres.

NB : Les indices des lignes et des colonnes de la matrice **M1** sont les lettres **A, B, C, D, E** et **F**.

Etape2 : Générer un message intermédiaire **msgi**, en concaténant les résultats du chiffrement de chaque caractère du message **msg**. Le résultat du chiffrement d'un caractère est la concaténation de l'indice de la ligne avec l'indice de la colonne de la case contenant le caractère à chiffrer.

Le caractère espace ne sera pas chiffré.

Etape3 : Remplir une deuxième matrice **M2** de taille **7x6** caractères en mettant dans :

- la première ligne, les lettres d'un mot-clé formé de 6 lettres majuscules,
- le reste des lignes, le message **msgi** caractère par caractère en commençant par la première case de la deuxième ligne.

NB : Chaque case vide de la matrice **M2** sera remplie par le caractère espace.

Etape4 : Trier les éléments de la 1^{ère} ligne de **M2** selon un ordre alphabétique croissant, sachant que tout déplacement d'un élément entraîne le déplacement de tous les éléments de la colonne correspondante.

Etape5 : Concaténer les lettres de la matrice **M2**, colonne par colonne en commençant par la 1^{ère} colonne et sans considérer les éléments de la 1^{ère} ligne, pour obtenir le message chiffré final.

Exemple :

Pour **msg** = "promotion bac 2019" et le mot-clé "CHAISE"

Etape1 : La matrice **M1** est remplie aléatoirement comme suit :

	A	B	C	D	E	F
A	c	1	o	f	w	j
B	y	m	t	5	b	4
C	i	7	a	2	8	s
D	p	3	0	q	h	x
E	k	e	u	l	6	d
F	v	r	g	z	n	9

Etape2 : Le résultat du chiffrement du message **msg**, caractère par caractère donne :

Caractère à chiffrer	p	r	o	m	o	t	i	o	n		b	a	c		2	0	1	9
Résultat du chiffrement	DA	FB	AC	BB	AC	BC	CA	AC	FE		BE	CC	AA		CD	DC	AB	FF

D'où le message **msgi** est le suivant : "DAFBACBBACBCCAACFE BECCAA CDDCABFF"

Etape3 : Remplissage de la matrice **M2**

C	H	A	I	S	E
D	A	F	B	A	C
B	B	A	C	B	C
C	A	A	C	F	E
	B	E	C	C	A
A		C	D	D	C
A	B	F	F		

Etape4 : Tri de la matrice **M2**

A	C	E	H	I	S
F	D	C	A	B	A
A	B	C	B	C	B
A	C	E	A	C	F
E		A	B	C	C
C	A	C		D	D
F	A		B	F	

Etape5 :

Le message chiffré final est "FAAECFDBC AACCEAC ABAB BBCCCDFABFCD "

On se propose d'écrire un programme permettant :

- de saisir un message **msg** ne dépassant pas 18 caractères et formé uniquement de lettres minuscules, de chiffres et d'espaces.
 - de saisir un mot-clé formé de 6 lettres majuscules.
 - de crypter le message **msg** selon la méthode de chiffrement décrite précédemment.
- NB :** On dispose d'un module **Initialisation(M1)** qui permet de remplir aléatoirement la matrice carrée **M1** comme décrit dans l'étape 1 et que le candidat peut appeler dans sa solution sans le développer.
- d'afficher le message chiffré final.

Travail demandé :

1. Analyser le problème en le décomposant en modules.
2. Ecrire les algorithmes des modules envisagés.