Ministère de l'éducation et de la formation **Direction régionale de Monastir** ꕥꕥꕥꕥꕥꕥ **Année Scolaire : 2008-2009**	Classes : 4ème Sciences de l'informatique
	Devoir de synthèse n°3 Algorithmique
	Date : Jeudi 8 Mai 2009
	Durée : 3 Heures

EXERCICE I (2 pts)

Soit la formule suivante de **pi** :

$$\pi = 2 * \sqrt{3} * \left(1 - \frac{1}{3} * \frac{1}{3^1} + \frac{1}{5} * \frac{1}{3^2} - \frac{1}{7} * \frac{1}{3^3} + ...\right)$$

Etablir l'algorithme du module qui permet de calculer la valeur approchée de π à $\mathbf{10^{-4}}$ prés.
Le calcul s'arrête quand la différence entre deux termes consécutifs devient inférieure ou égale à 10^{-4}.

EXERCICE II (4 pts)

On se propose de chercher un entier **X** donné dans un tableau **T** de **n** entiers (le tableau est trié en ordre croissant) en utilisant la méthode de recherche trichotomique.
Le principe de cette méthode est décrit comme suit :

1- On compare l'entier à chercher **X** avec **T[p1]** et **T[p2]**
2- Si **X** est **égale à l'un de deux, la recherche est terminée**, sinon **s'il est inférieur à T[p1] on refait la recherche dans la partie gauche du tableau** qui réside avant t[p1], sinon **s'il est inférieur à T[p2] on refait la recherche dans la partie du milieu** du tableau qui réside entre p1 et p2, sinon **on refait la recherche dans la partie droite du tableau** qui réside après T[p2] .

Sachant que **P1=(2*D+F) Div 3** et **P2=(D+2*F) Div 3** où **D** et **F** sont respectivement les indices du début et de la fin de la partie du tableau dans laquelle on va continuer la recherche.

Travail à réaliser :

Etablir l'algorithme du module recherche « trichotomique » en utilisant une méthode récursive.

PROBLEME (14 Pts)

On se propose d'écrire un programme qui permet de corriger les fautes d'orthographe d'un fichier texte «**C:\bac2009\docum.txt** » puis de l'afficher. Ce fichier renferme plusieurs lignes dont chacune renferme une phrase ; sachant que chaque phrase est une suite des mots et deux mots successifs sont séparés par un seul espace.

La correction se fait en se referant à un fichier dictionnaire suivant «**C:\bac2009\dico.dic** » qui renferme des mots de **25** caractères au maximum.

Le processus de correction se déroule comme suit :

On consulte le fichier texte à corriger puis on vérifie l'existence de chacun de ses mots dans le dictionnaire. Si un mot n'existe pas alors un menu sera affiché présentant les choix suivants :

1- **Ignorer l'erreur** : dans ce cas le mot sera conservé tel qu'il est
2- **Ajouter au dictionnaire** : dans ce cas le mot sera ajouté à la fin du dictionnaire.
3- **Suggestion** : le programme détermine et affiche une liste de **5 mots au maximum** qui ont un degré de ressemblance au mot à corriger > à 75%. si la liste de suggestion est vide alors le programme affichera le message « **aucune suggestion** » sinon l'utilisateur donne l'indice du mot à utiliser à partir de la liste des mots suggérés pour corriger le fichier texte.

On définit le degré de ressemblance entre deux mots par :

$DR = (NCCBP^2 + NCCMP)/(L1*L2)*100$

Avec :

NCCBP : nombre de caractères communs et bien placés

NCCMP : nombre de caractères communs et mal placés

L1, L2 : sont respectivement les longueurs du mot1 et du mot2

Exemples :

Exemple 1:	**Exemple 2 :**
Mot1='Ecol'	Mot1='abbcda'
Mot2='Ecole'	Mot2='acbdx'
NCCBP=4	NCCBP=2
NCCMP=0	NCCMP=2
L1=4	L1=6
L2=5	L2=5
$DR=(4^2+0)/(4*5)*100=16/20*100=80$ %	$DR=(2^2+2)/(6*5)*100=6/30*100=20$ %

N.B. : On ne peut pas modifier une ligne dans un fichier texte ; on peut utiliser un fichier texte intermédiaire.

Travail demandé :

1- Donner les structures des données nécessaires pour résoudre ce problème.
2- Analyser le problème en le décomposant en modules.
3- Analyser les différents modules envisagés

Correction devoir de synthèse N°3

Problème

I. Structure de données (1 point)

Pour résoudre ce problème on a besoin de :

- Un fichier texte pour traiter le fichier à corriger
- Un fichier typé pour le fichier dictionnaire

II. Programme principal {2 points : analyse et TDO}

Analyse

	Nom : DevSynthes3	
S	LDE	OU
4 3 2 1 5	Résultat=proc affiche ($FichText_f$) $FichText_f$= proc corriger($fichtext_i$, Fdic) $Fichtext_i$, = associer ($fichtext_i$,"C:\bac2009\docum.txt") FDic= associer(Fdic, " C:\bac2009\dico.dic") Fin DevSynthese1	$FichText_f$ $Fichtext_i$ Fdic

Objets	Type/Nature	Rôle
$Fichtext_{(f,i)}$	Texte	Fichier texte à corriger (i et f représente l'état du fichier avant et après la correction.
Fdic	TFdic	Fichier contenant les mots du dictiobnnaire
affiche	procédure	Affiche le contenu du fichier s'il a subi une correction
Corriger	procédure	Corrige les fautes d'orthographe du fichier texte.

TDNT

Type
TFDic= fichier de chaine[26]

III. modules {11 points}

1. Analyse procédure affiche {1 points}

	Def Proc affiche (var f :texte)	
S	LDE	OU
3 2 1 4	Résultat= affichage Affichage= [Ouvrir (f)] Tant Que non finfichier(f) faire Lire_nl(f,ligne) Ecrire_nl(ligne) Fin tant que Fermer(f) Fin si Fin affichage	ligne

Objets	Type/Nature	Rôle
ligne	Chaine	Contient une ligne du fichier texte

2. Analyse de la Pprocédure corriger {1.5 point}

	Def proc corriger(var f :texte ;var fdic :)	
S	LDE	OU
1 2	Résultat= f F= [associer(ftemp,c:\bac2009\temp.txt') ; réécrire (ftemp)] Tantque non finfichier(f) faire Lire(f,ligne) Ligne←Fn corrigeligne(ligne,fdic) Ecrire(ftemp, ligne) Fin tantque Fermer (ftemp) Efface(f) Renommer (ftemp, ,"C:\bac2009\docum.txt") Fin si Fin corriger	Corrigeligne ftemp

objets	Type/Nature	Rôle
corrigeligne	fonction	Corrige une ligne
Ftemp	texte	Fichier temporaire

3. Analyse de la fonction corrigeligne (2,5 points)

	Def Fn corrigeligne(ch :chaine var fdic :TFDic) :chaine	
S	LDE	OU
2	Résultat=CorrigeLigne←ChCor	Mot
1	ChCor=[chcor←'' ;Insere(' ',ch,long(ch)+1)]	Choix
	P←pos(' ',ch)	J
	Tantque p<> faire	n
	Mot← SousChaine(ch,1,p-1)	listemots
	Si non Fn existe (mot, fdic) alors	
	Ecrire ('1- Ignorer')	
	Ecrire ('2- Ajouter au dictionnaire')	
	Ecrire ('3- Suggestion')	
	Répéter	
	Choix ← liretouche { ou choix=donnée	
	Jusqu'à choix dans ['1'.. '3']	
	Selon choix faire	
	'2' : Proc ajouter (mot, fdic)	
	'3' : Proc suggerer (mot, listemots, n, fdic)	
	P← i-long(mot)	
	Si n = 0 alors	
	Ecrire ('pas de sugestion')	
	Sinon	
	Pour j de 1 à n faire	
	Ecrire(listemots[j])	
	Fin pour	
	Répéter	
	Lire(j)	
	Jusqu'à j dans [0..n]	
	Si i>0 alors	
	Mot :=listemots[j]	
	Fin si	
	Fin si	
	Fin selon	
	Fin Si	
	chcor←chcor+mot	
	efface(ch,1,p)	
	p ←pos(' ',ch)	
	fin tantque	
3	Fin corrigeligne	

Objets	Type/Nature	Rôle
j	entier	compteur
Mot	chaine	Mot à corriger
Choix	caractère	Choix de l'utilisateur
n	entier	Nombre de mot à suggérer
listemots	Tab2	Liste des mots à suggérer

Type
Tab2= tableau de 5 chaine [26]

4. Analyse procédure ajouter {1 point}

	Def proc ajouter (mot :chaine ; var fdic : tfdic)	

S	LDE	OU
2 1 3	Résultat=fermer(fdic) fdic=[ouvrir (fdic)] pointer(fdic,tailleFichier(fdic)) ecrire(fdic,mot) Fin ajouter	

5. Analyse procédure suggerer {2 point}

	Def proc suggerer (mot :chaine ; var liste :tab2 ; var n : entier ;var fdic : tfdic)	
S	LDE	OU
1 1	Résultat=Liste,n Liste,n=[ouvrir (fdic) ;n←0] Tantque non FinFichier(fdic) et(n <5) faire Lire(fdic,m) Si Fn DegRes(mot,m) >75 alors n←n+1 Liste[n]← m Fin si Fin tantque Fin suggerer	m DegRes

Objets	Type/Nature	Rôle
m	chaine	Le mot lu du dictionnaire
DegRes	Fonction de type entier	Calcule le degré de ressemblance entre deux mots

6. Analyse Fonction Existe (1 point)

	Def Fn existe (mot :chaine var f : tfdic)	
S	LDE	OU
2 1 3	Résultat=Existe ← trouver M=[ouvrir (fdic) ;trouver ←faux] Tantque non FinFichier(fdic) et(non trouver) faire Lire(fdic,m) Trouver←mot=m Fin tantque Fin suggerer	m trouver

Objets	Type/Nature	Rôle
m	chaine	Le mot lu du dictionnaire
trouver	boolean	Le mot est trouvé ou non dans le dictionnaire

7.

Analyse fonction DegRes {2 points}

	Def fn DegRes(m1,m2 :chaine) :entier	
S	LDE	OU
4	Résultat= DegRes←(NCCBP²+NCCMP)/(L1*L2)*100	NBCCBP
1	L1,L2=[L1←long(m1) ;l2←long(m2)]	NCCMP
	Si l1>l2 alors	L1
	m← m1	L2
	m1←m2	L
	m2←m	m
	l←l1	
	l1←l2	
	l2← l	
	Fin si	
2	NCCBP=[NCCBP←0]	
	Pour i de 1 à l1 faire	
	Si majus(m1[i])=majus(m2[i]) alors	
	NCCBP←NCCBP+1	
	M1[i]←' '	
	M2[i]←' '	
	Fin si	
	Fin pour	
3	NCCMP=[NCCMP←0	
	pour i de 1 à l2 faire	
	m2[i]← Majus(m2[i])	
	fin pour]	
	Pour i de 1 à l1 faire	
	P←pos(majus(m1[i]),m2)	
	Si (p<> 0) et m1[i]<>' ' alors	
	NCCMP← NCCMP+1	
5	M2[p]←' '	
	Fin si	
	Fin Pour	
	Fin DegRes	

Objets	Type/Nature	Rôle
NBCCBP	entier	Nombre de caractères communs et bien placés
NCCMP	entier	Nombre de caractères communs et mal placés
L1	entier	Longueur du mot le plus court
L2	entier	Longueur du mot le plus long
m	chaine	Variable intermédiaire
l	entier	Variable intermédiaire