

Première partie : (10 points)

Exercice n°1 : (3 points)

- 1- On rappelle que pour que la boucle **tantque** soit exécutée, la condition doit être **vérifiée**. Dans notre cas, la valeur de COMPTEUR doit être inférieure à 4.

V 0

V 3

F 4

F 5

- 2- Pour que le traitement s'exécute exactement **10 fois** et compte tenu que l'avancement dans le compteur se fait par **un pas=1**, la valeur de X est déterminée de la manière suivante :

$$\text{Nombre de répétitions} = (\text{Valeur finale} - \text{valeur initiale}) + 1$$

$$10 = 3 - X + 1$$

$$X = 3 + 1 - 10 = -6$$

V -6

F -4

F 6

F 10

- 3- La trace du programme révèle les valeurs suivantes :

Somme	Compteur	CA
0	0	0
0	1	10
10	2	20
30	3	30
60	4	

F 0

F 30

F 40

V 60

Exercice n°2 : (3,5 points)

Dans cet exercice il est demandé l'**algorithme** d'une **fonction**.

D'après la définition de la suite, on déduit qu'elle est définie selon 2 cas :

- 1) Cas 1, $n=0$, le résultat (U) est égal à $1 + \frac{1}{n}$, puisque la valeur du résultat est fixée, il s'agit d'un **cas d'arrêt** du traitement récursif.

- 2) Cas 2, $n \geq 1$, le résultat (U) est égal à $1 + \frac{1}{U_{n-1}}$, puisque la valeur du résultat n'est pas fixée et est définie en fonction de U_{n-1} , il ne s'agit pas d'un cas d'arrêt du traitement. De plus, comme le résultat est défini en fonction du terme précédent, on en déduit qu'il s'agit de la définition de l'étape **d'avancement dans le traitement récursif**

On obtient alors l'algorithme suivant pour la **fonction Calc-Suite** :

1. Calcul du $n^{\text{ième}}$ terme de U

Solution récursive

0/ **DEFFN** Calc_Suite (n , m: entier) : réel

1/ Si $n = 0$ Alors $\text{Calc_Suite} \leftarrow 1 + 1/m$
 Sinon $\text{Calc_Suite} \leftarrow 1 + 1/ \text{FN Calc_Suite}(n-1,m)$
 Finsi

2/ **FIN** Calc_Suite

Solution itérative

0/ **DEFFN** Calc_Suite (n , m: entier) : réel

1/ $s \leftarrow 1 + 1/m$

Pour i de 1 à n faire

$s \leftarrow 1 + 1/s$

FinPour

2/ $\text{Calc_Suite} \leftarrow s$

3/ **FIN** Calc_Suite

2. L'ordre de récurrence de la fonction est égal à 1 car le terme U_n est **défini à partir du terme** U_{n-1} .

Exercice n°3 : (3,5 points)

Analyse de la **fonction Nb_Lig_Sym** permettant de déterminer le nombre de lignes symétriques de la matrice **Mat** :

DEFFN Nb_Lig_Sym (Mat : matrice; m, n : entier) : entier

Résultat = Nb_Lig_Sym \leftarrow nb

nb = [nb \leftarrow 0]

Pour L de 1 à m Faire

Si FN Sym (Mat,L,n)

Alors nb \leftarrow nb+1

FinSi

FinPour

Fin Nb_Lig_Sym

TDO Locaux

Objet	Type/Nature	Rôle
nb	Entier	Nombre de lignes symétriques
L	Entier	Compteur
Sym	Fonction/Booléen	Fonction permettant de vérifier la symétrie d'une ligne

DEFFN Sym (Mat : matrice ; lig, n : entier) : Booléen

Résultat = Sym \leftarrow valid

Valid = [i \leftarrow 1, valid \leftarrow (Mat[lig,i]=Mat[lig,n])]

Tant que (i<=n Div 2) Et valid Faire

i \leftarrow i+1

valid \leftarrow (Mat[lig,i]=Mat[lig,n-i+1])

FinTant que

Fin Sym

TDO Locaux

Objet	Type/Nature
i	Entier
valid	Booléen/logique

Deuxième partie : (10 points)

Analyse du programme principal

DEBUT **Cryptage**

Résultat = TR

TR = Proc Conv_nbr_txt (TR, MC_O, lig, Long_Max)

MC_O = Proc Conv_Oct (MC_D, lig, Long_Max)

MC_D = Proc Remplir (TD, MC_D, lig, Long_Max)

TD = Assigner (TD, "C:\txtinit.txt")

lig ← FN Nb_ligne (TD)

Long_Max ← FN Plus_long (TD)

Fin **Cryptage**

Tableau de déclaration des nouveaux types

Type
Matrice = Matrice [1..50, 1..50] d'entiers

Tableau de déclaration des objets globaux

Objet	Type/Nature	Rôle
TD	Texte	Texte à crypter (supposé déjà saisi)
TR	Texte	Texte résultat (texte crypté)
MC	Matrice	Matrice contenant les codes ascii des caractères de T, puis convertis de la base 10 vers la base 8
Col	Octet	Nombre de colonnes de MCD (nombre de caractères dans la ligne la plus longue de T)
Lig	Octet	Nombre de lignes de MCD (nombre de lignes dans T)
Plus_long	Fonction	Fonction qui retourne la valeur de Col
Nb_ligne	Fonction	Fonction qui retourne la valeur de Lig
Conv_Oct	Procédure	Procédure qui convertit les éléments de MCD en octal
Conv_nbr_txt	Procédure	Procédure qui remplit le fichier texte TR à partir de MCD
Remplir	Procédure	Procédure qui remplit la matrice MC à partir du fichier texte T

Analyse de la procédure Remplir

Rôle du module : Remplissage à partir du fichier texte TD de la matrice M, qui contiendra, soit le code ASCII du caractère à traiter $M[L, c] \leftarrow \text{Ord}(\text{ph}[c])$ soit un espace ($M[L, c] \leftarrow \text{Ord}(" ")$).

DEFPROC **Remplir** (Var TD : Texte ; Var M : Matrice ; Lig, Long_Max : Octet)

Résultat = M

M = [L ← 0, Ouvrir (TD)]

Tant que (Non Fin_fichier(TD)) **Faire**

L ← L + 1

Lire (TD, ph)

Pour c De 1 à Long_Max **Faire**

Si (c ≤ Long (ph)) **Alors** M[L, c] ← Ord(ph[c])

Sinon M[L, c] ← Ord(" ")

FinSi

FinPour

FinTant que

• Traitement de toutes les lignes du fichier

• Toutes les lignes ont la même longueur **Long_Max**

• Traitement des **premiers** caractères c de chaque ligne L de TD. Astuce : on utilise les mêmes compteurs pour la matrice M.

• Traitement des **derniers** caractères (si nécessaire) : Ajout des espaces pour obtenir des lignes de même longueur. **FinTant que**

Fin Remplir

Tableau de déclaration des objets locaux

Objet	Type/Nat	Rôle
L	Octet	Compteur pour les lignes
c	Octet	Compteur pour les colonnes
ph	Chaîne	Variable contenant chaque ligne lue du fichier

Analyse de la procédure Conv_nbr_txt

Rôle du module : Remplissage à partir de la matrice M, du fichier texte TR.

DEFPROC Conv_nbr_txt (Var TR : Texte ; MC : Matrice ; lig, Long_Max : Octet)

Résultat = TR

TR = [Assigner (TR, "C:/txtcryp.txt"), Recréer (TR)]

Pour c De 1 à Long_Max Faire

Ph ← ""

Pour L De 1 à lig Faire

Convch (MC [L,c], ch)

Ph ← ph + ch + " "

Finpour

Ecrire_nl (TR, ph)

Finpour

Fin Conv_nbr_txt

- Traitement de **toutes** les colonnes de la matrice **M** (parcours **vertical** de M)
- Traitement de **toutes** les lignes de la matrice **M** (parcours **horizontal** de M)
- Conversion de chaque élément de la matrice en caractère.
- Rangement de chaque résultat intermédiaire **ch**, dans la variable **Ph** et séparation des résultats intermédiaires par un **espace**.

Tableau de déclaration des objets locaux

Objet	Type/Nature	Rôle
L	Octet	Compteur pour les lignes
c	Octet	Compteur pour les colonnes
ph	Chaîne	Variable contenant chaque ligne lue du fichier
ch	Chaîne	Variable contenant la conversion d'un élément de la matrice en chaîne

Analyse de la procédure Conv_Oct

Rôle du module : Conversion (cryptage) du contenu de la matrice M. Le cryptage consiste à convertir en base 8

Conv10_8 (M[L,c]) chaque caractère de la matrice et le ranger dans la même case.

DEFPROC Conv_Oct (Var M : Matrice ; Lig, Long_Max : Octet)

Résultat = M

M = []

Pour L De 1 à lig Faire

Pour c De 1 à Long_Max Faire

M[L,c] ← FN Conv10_8 (M[L,c])

FinPour

FinPour

Fin Conv_Oct

- Traitement de toutes les lignes de la matrice M (parcours horizontal de M)
- Traitement de toutes les colonnes de la matrice (parcours vertical de M)
- Conversion de chaque élément de la matrice en base 8.

Tableau de déclaration des objets locaux

Objet	Type/Nature	Rôle
L	Octet	Compteur pour les lignes
c	Octet	Compteur pour les colonnes
Conv10_8	Fonction	Fonction qui retourne la conversion d'entier de la base 10 vers la base 8

Analyse de la fonction Conv10_8

Rôle du module : Conversion d'un entier **d** en base **8**

Def FN **Conv10_8** (d : Entier) : Entier

Résultat = Conv10_8 ← Res

Res = Valeur (ch, Res, e)

Ch = [Ch ← ""]

Répéter

R ← d MOD 8

Convch (R, Ch1)

d ← d div 8

Ch ← Ch1 + Ch

Jusqu'à (d = 0)

Fin **Conv10_8**

Tableau de déclaration des objets locaux

Objet	Type/Nature	Rôle
R	Entier	Variable pour stocker le reste de la division
Res	Entier	Variable pour stocker le résultat de la fonction
e	Entier	Valeur de l'erreur
Ch1	Chaîne	Variable pour stocker la conversion de R en chaîne
Ch	Chaîne	Chaîne résultat de la conversion

Analyse de la fonction Plus_long

Rôle du module : Parcours du fichier texte T et détermination de la ligne la plus longue (**Max**).

DEFFN **Plus_long** (Var T : Texte) : Octet

Résultat = Plus_long ← Max

Max = $\left[\begin{array}{l} Ouvrir(T) \\ Lire(T, ph) \\ Max \leftarrow Long(ph) \end{array} \right]$

Tant que (Non Fin_fichier (T)) **Faire**

Lire (T, ph)

Si (Long (ph) > Max) **Alors**
Max ← Long (ph)

Finsi

FinTant que

Fin **plus_long**

• Parcours de toutes les lignes du fichier texte **T**.

• Recherche de la longueur de phrase la plus longue et conservation de cette valeur dans la variable **Max**.

Tableau de déclaration des objets locaux

Objet	Type/Nature	Rôle
Max	Octet	Variable temporaire pour stocker le résultat
Ph	Chaîne	Variable pour stocker chaque ligne lue à partir du texte

Analyse de la fonction Nb_ligne

Rôle du module : Parcours du fichier texte **TD** et détermination de son nombre de lignes (**N**).

```

DEFFN Nb_ligne (Var TD : Texte) : Octet
Résultat = Nb_ligne ← N
N = [N ← 0, Ouvrir (TD)]

```

```

[ Tant que (Non Fin_fichier (TD)) Faire ]

```

```

Lire_nl (TD)

```

```

[ N ← N + 1 ]

```

- Parcours de toutes les lignes du fichier texte **TD**.

- Détermination du nombre de lignes du fichier **TD** et conservation de cette valeur dans la variable **N**.

FinTant que

Fin Nb_ligne

Tableau de déclaration des objets locaux

Objet	Type/Nature	Rôle
N	Octet	Variable temporaire pour stocker le résultat

Algorithme du programme principal

- 0) Début **cryptage**
- 1) Assigner (T, "C:\txtinit.txt")
- 2) Lig ← FN Nb_ligne (T)
- 3) Col ← FN Plus_long (T)
- 4) Proc Remplir (T, MC, lig, col)
Proc Conv_Oct (MC, lig, col)
- 5) Assigner (TR, "C:\txtcryp.txt"), Recréer (TR)
Proc Conv_nbr_txt (TR, MC, col, lig)
- 6) Fermer (T), Fermer (TR)
- 7) Fin **cryptage**

Algorithme de la procédure Remplir

- 0) DEFPROC **Remplir** (Var T : Texte ; Var M : Matrice ; Lig, Col : Octet)
- 1) L ← 0, Ouvrir (T)
 - Tant que** (Non Chercher_fin_fichier(T)) **Faire**
 - L ← L + 1
 - Lire (T, ph)
 - Pour** c De 1 à col **Faire**
 - Si** (c ≤ Long (ph) Alors M[L,c] ← Ord(ph[c])
 - Sinon M[L,c] ← Ord(" ")
 - FinSi**
 - FinPour**
 - FinTant que**

2) Fin Remplir

Algorithme de la procédure Conv_nbr_txt

0) DefProc Conv_nbr_txt (Var TR : Texte ; MC : Matrice ; col, lig : Octet)

1) Ajouter (TR)

Pour L De 1 à lig Faire

 Ph ← ""

Pour c De 1 à col Faire

 Convch (MC[L,c], ch)

 Ph ← ph + ch

Finpour

 Ecrire_nl (TR, ph)

Finpour

2) Fin Conv_nbr_txt

Algorithme de la procédure Conv_Oct

0) DEFPROC Conv_Oct (Var M : Matrice ; Lig, Col : Octet)

1) **Pour** L De 1 à lig Faire

Pour c De 1 à col Faire

 M[L,c] ← FN Conv10_8 (M[L,c])

FinPour

FinPour

2) Fin Conv_Oct

Algorithme de la fonction Conv10_8

0) Def FN Conv10_8 (d : Entier) : Entier

1) Ch ← ""

Répéter

 R ← d MOD 8

 Convch (R, Ch1)

 d ← d div 8

 Ch ← Ch1 + Ch

Jusqu'à (d = 0)

2) Valeur (ch, Res, e)

3) Conv10_8 ← Res

4) Fin Conv10_8

Algorithme de la fonction Nb_ligne

0) Def FN Nb_ligne (Var T : Texte) : Octet

1) N ← 0, Ouvrir (T)

Tant que (Non Fin_fichier (T)) Faire

 Lire_nl (T)

 N ← N + 1

FinTant que

2) Nb_ligne ← N

3) Fin Nb_ligne

Algorithme de la fonction plus_long

0) Def FN Plus_long (Var T : Texte) : Octet

1) Ouvrir (T)

 Lire (T, ph)

Max \leftarrow Long (ph)
Tant que (Non Fin_fichier (T)) Faire
 Lire (T, ph)
 Si (Long (ph) > Max) Alors
 Max \leftarrow Long (ph)
 Finsi
FinTant que
2) Plus_long \leftarrow Max
3) Fin plus_long