

**Examen du baccalauréat 2014**  
**Corrigé de l'épreuve d'Informatique**  
**Mathématiques – Sciences Expérimentales – Sciences Techniques**

**Exercice 1 : (3.5 points = 5\*0.5 + 2\*0.5)**

| Instruction                                                   | Vrai/ faux | Justification                                                                                              |
|---------------------------------------------------------------|------------|------------------------------------------------------------------------------------------------------------|
| $a \leftarrow \text{ORD}(\text{SUCC}(\text{Vendredi}))$       | Vrai       |                                                                                                            |
| $T[\text{Vendredi}] \leftarrow \text{"Bonjour"}$              | Vrai       |                                                                                                            |
| $\text{Ch}[15] \leftarrow \text{sous-chaîne ("Bac",1,1)}$     | Faux       | Il est impossible d'affecter une chaîne à un caractère.<br>On acceptera aussi $\text{Ch}[15]$ n'existe pas |
| Lire (j)                                                      | Faux       | On ne peut pas lire une variable de type scalaire énuméré.                                                 |
| Pour i de Lundi à Vendredi faire<br>Ecrire (T[i])<br>Fin pour | Vrai       |                                                                                                            |

**Exercice 2 : (4.5 points = 4\*0.5 + 1.5+1)**

1) Pour chacune des propositions suivantes mettre dans la case correspondante la réponse "**Vrai**" si l'entête de la fonction **Pythagore** est correcte ou la réponse "**Faux**" dans le cas contraire.

| Entête proposée pour la définition de la fonction Pythagore | Réponse |
|-------------------------------------------------------------|---------|
| DEF FN Pythagore (a,b,c : Entier ) : booléen                | Faux    |
| DEF FN Pythagore (n1,n2,n3 : réel ) : booléen               | Faux    |
| DEF FN Pythagore (a,b,c : Réel ) : booléen                  | Vrai    |
| DEF FN Pythagore (a,b,c : Réel): réel                       | Faux    |

2) L'élève a oublié l'appel de la fonction **Pythagore** dans l'algorithme du programme appelant, Réécrire l'algorithme **Prg\_Appelant** en ajoutant à l'endroit convenable l'appel adéquat de cette fonction.

0) Début Prg\_Appelant

1) Lire (n1)

2) Lire (n2)

3) Lire (n3)

4)  $R \leftarrow \text{FN Pythagore}(n1,n2,n3)$

5) Si R=Vrai Alors

Ecrire (n1, ",", n2, "et ", n3, "forment un triplet de Pythagore ")

Finsi

6) Fin Prg\_Appelant

3) Améliorer l'instruction numéro 1) de la fonction **Pythagore** en remplaçant la structure de contrôle conditionnelle par une structure simple.

**Pythagore**  $\square \text{carre}(a) + \text{carre}(b) = \text{carre}(c)$

## Problème : (12 points)

### 1) Analyse du Programme Principal

**Nom :** cryptage

**Résultat** =Ecrire("Le message crypté est ", FN Crypter (msg))

msg=PROC Saisie(msg)

**Fin cryptage**

| Tableau de déclaration des Objets | Objet | Type/Nature | Rôle                                    |
|-----------------------------------|-------|-------------|-----------------------------------------|
| msg                               |       | Chaîne      | Chaîne à crypter                        |
| Crypter                           |       | Fonction    | Permet le cryptage de la chaîne msg     |
| saisie                            |       | Procédure   | Permet la saisie de la chaîne à crypter |

### 2) Analyse des modules :

➤ **Analyse de la procédure saisie :**

**DEF PROC** saisie (var msg :chaîne )

**Résultat** =msg

msg= [ ] **Répéter**

msg= donnée("Saisir le message a crypté ")

**Jusqu'à** (pos ("\_",msg)=0) et (FN verif (msg) =vrai)

**Fin saisie**

**T.D.O.L**

| Nom   | Type     | Rôle                                                            |
|-------|----------|-----------------------------------------------------------------|
| verif | fonction | Vérifier l'existence des signes de ponctuation dans le message. |

➤ **Analyse de la fonction verif :**

**DEF FN** verif (Ch : chaîne) : booléen

**Résultat** = verif ← R

R= [i←0] **Répéter**

i←i+1

**jusqu'à** (ch[i] dans [",", ".", " ", "!", "?", ":"]) ou (i= long(ch))

**Si** (ch[i] dans [",", ".", " ", "!", "?", ":"]) **Alors** verif← faux

**Sinon** verif← vrai

**Fin si**

**Fin Verif**

**T.D.O.L**

| Nom | Type   | Rôle     |
|-----|--------|----------|
| I   | Entier | Compteur |

➤ **Analyse de la fonction Crypter :**

**DEF FN** Crypter (msg : chaîne) : chaîne

**Résultat** = Cypter ← Ch

Ch = [Ch←""] **Pour i de 1 à lmax faire**

Ch←Ch + T1[i]+ "\_"

**Fin pour**

Ch←sous-chaîne(ch,1,long(ch)-1)

T1= PROC repartir (T, n, T1, lmax)

(T,n,lmax) =PROC separer (msg, T, n)

PROC ajoutetoile (T, n,lmax)

**Fin Crypter**

### Tableau de déclaration de Nouveaux Types

| Types                        |
|------------------------------|
| vect = Tableau de 20 chaines |

### T.D.O.L

| Nom         | Type      | Rôle                                                         |
|-------------|-----------|--------------------------------------------------------------|
| i           | Entier    | Compteur                                                     |
| lmax        | Entier    | longueur du plus long mot dans le tableau T                  |
| n           | Entier    | Le nombre de mots extrait de Ch c'est la taille du tableau T |
| Ch          | Chaine    | Le message crypté                                            |
| T           | Vect      | Contient les mots de Ch après sa décomposition               |
| T1          | Vect      | Contient l'éclatement des chaines du tableau T               |
| repartir    | Procédure | Permet d'éclater les chaines de T dans T1                    |
| separer     | Procédure | Permet de décomposer la chaine Ch en des mots                |
| ajoutetoile | Procédure | Permet d'ajouter des étoiles                                 |

#### ➤ Analyse de la procédure repartir:

**DEF PROC Repartir** (T : vect ; n : entier ; var T1 : vect ; lmax : entier )

**Résultat = T1**

**T1=[]**

**Pour i de 1 à lmax faire**

**T1[i] ← ""**

**Pour j de 1 à n faire**

**T1[i] ← concat (t1[i] ,t[j][i])**

**Fin pour**

**Fin pour**

**i , j =compteur**

**Fin repartir**

### T.D.O.L

| Objet | Type/Nature | Rôle     |
|-------|-------------|----------|
| i     | Entier      | Compteur |
| j     | Entier      | Compteur |

#### ➤ Analyse de la procédure separer :

**DEF PROC separer** (ch :chaine ; Var T : vect ; var n :entier )

**Résultat =T,n**

**(t,n) =[n←0]**

**Tant que** (pos(" ",ch)≠0) **faire**

**n←n+1**

**T[n] ←sous-chaine (ch,1,pos(" ",ch)-1)**

**Efface (ch,1,pos(" ",ch))**

**Fin tant que**

**n←n+1**

**T[n] ←ch**

**Fin separer**

➤ **Analyse de la procédure ajoutetoile :**

**DEF PROC ajoutetoile**(Var T : vect ; n : entier ; var lmax : entier )

**Résultat =T,lmax**

lmax=[lmax←long(T[1] ]

**Pour i de 2 à n faire**

    Si (long(t[i])>lmax) Alors

        lmax←long(t[i])

    Fin si

**Fin pour**

**T= [ ]**

**Pour i de 1 à n faire**

**Tant que** (long(T[i]) <lmax) **faire**

        T[i]← T[i] + "\*"

**Fin tant que**

**Fin pour**

**Fin ajoutetoile**

**T.D.O.L**

| Objet | Type/Nature | Rôle     |
|-------|-------------|----------|
| i     | Entier      | Compteur |

