RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION ***** EXAMEN DU BACCALAURÉAT	Épreuve : **INFORMATIQUE**	
	Sections : **Mathématiques, Sciences Expérimentales et Sciences Techniques**	
	Durée : **1 H 30**	Coefficient : **0.5**
SESSION 2016		

Le sujet comporte 5 pages numérotées de 1/5 à 5/5.
Les réponses aux exercices 1 et 2 doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve

Exercice 1 : (3 points)

Valider chacune des propositions ci-dessous en mettant dans la case correspondante la lettre (**V**) si elle est correcte ou la lettre (**F**) dans le cas contraire.

1) L'identificateur d'une variable :

- ☐ ne doit pas commencer par un chiffre.
- ☐ peut contenir un espace.
- ☐ peut contenir le caractère souligné (tiret bas "_").

2) L'instruction qui permet d'affecter à une variable **X**, une valeur aléatoire de l'intervalle **[2,10]** est :

- ☐ X ← 2 + Aléa (10)
- ☐ X ← 2 + Aléa (9)
- ☐ X ← Aléa (2 + 10)

3) Une structure de données tableau peut :

- ☐ contenir des éléments de types différents.
- ☐ être déclarée avec une taille maximale variable.
- ☐ avoir des indices de cases de type caractère.

4) Pour le type scalaire énuméré :

- ☐ les valeurs énumérées peuvent appartenir à un type prédéfini.
- ☐ une valeur énumérée peut être affectée à une variable du même type.
- ☐ les opérateurs relationnels "<", ">" et "=" peuvent être appliqués.

Exercice 2 : (5 points)

Soient les algorithmes ci-dessous correspondant à un programme principal **Exercice** et à une fonction **Inconnue** appelée par celui-ci :

```
0) Début Exercice
1) Lire (A)
2) Si FN Inconnue (A) Alors
        Ecrire (A, " Vérifie la propriété.")
   Sinon
        Ecrire (A, " Ne vérifie pas la propriété.")
   Fin Si
3) Fin Exercice
```

```
0) Def FN Inconnue (C : .....) : .....
1) Répéter
        Valeur (C[1], X, E)
        Efface (C, 1, 1)
   Jusqu'à (C = "" ) ou (E ≠ 0)
2) Inconnue ← E = 0
3) Fin Inconnue
```

1) A partir des algorithmes donnés ci-dessus, remplir la 2ème colonne du tableau suivant par un exemple de chaque élément cité dans la 1ère colonne :

Elément	Exemple
Expression booléenne	.....
Procédure prédéfinie	.....
Paramètre formel	.....
Paramètre effectif	.....

2) Compléter l'entête de la fonction **Inconnue** par les types appropriés.

Def Fn **Inconnue** (C : .....) : .....

3) Compléter le tableau de déclaration ci-dessous par les types des objets locaux de la fonction **Inconnue**.

Objet	Type / Nature
X	.....
E	.....

4) Parmi les variables **A, C, X** et **E**, réécrire dans le tableau ci-dessous celles qui ne sont pas visibles par le programme principal.

Variables non visibles par le programme principal
.....

5) Donner le résultat affiché par le programme **Exercice** pour chacune des valeurs de la variable **A** suivantes :

- A = "523" ➔ .....
- A = "-523" ➔ .....
- A = "5.23" ➔ .....
- A = "A5B3" ➔ .....

6) En déduire le rôle de la fonction **Inconnue**.

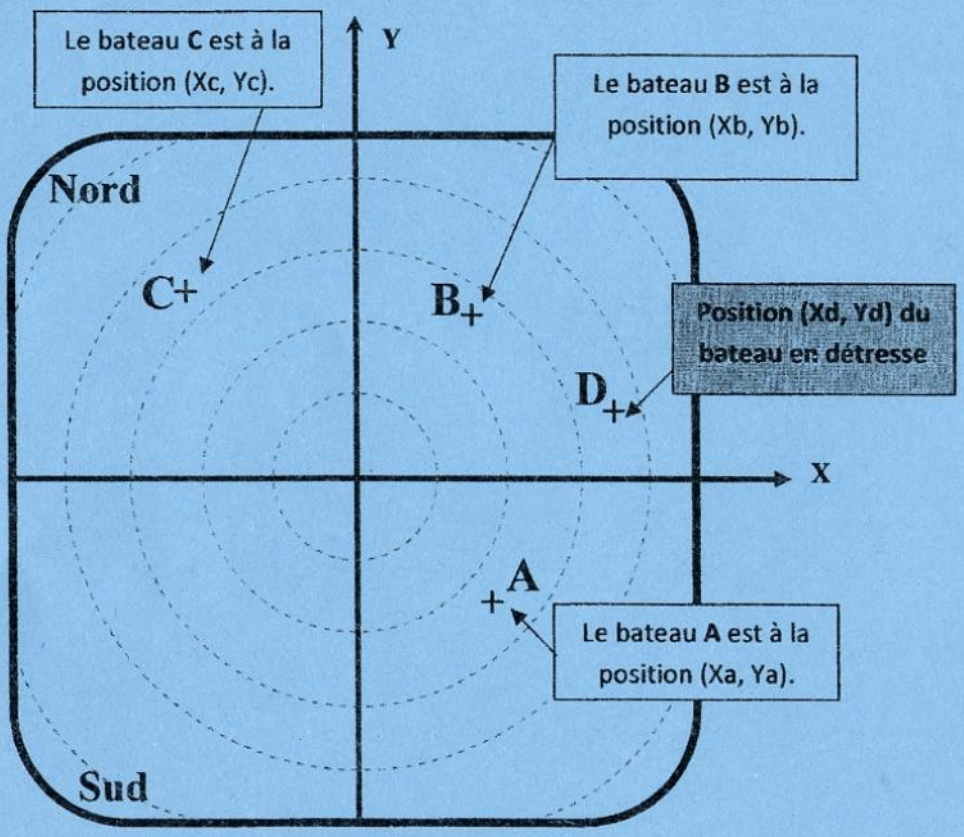
.....

.....

RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION ***** EXAMEN DU BACCALAURÉAT	Épreuve : **INFORMATIQUE**	
	Sections : **Mathématiques, Sciences Expérimentales et Sciences Techniques**	
	Durée : **1 H 30**	Coefficient : **0.5**
SESSION 2016		

Problème : (12 points)

Un bateau en détresse a lancé un appel de secours (SOS). Pour le sauver, le commandant de la garde côte a besoin de localiser le(s) bateau(x) proche(s) de celui-ci. En s'appuyant sur leurs coordonnés **(X, Y)** fournis par le radar du commandant (comme l'illustre l'exemple de la figure ci-dessous), la localisation des bateaux se fait par le calcul des distances qui les séparent du bateau en détresse.



Pour aider le commandant de la garde côte, on se propose d'écrire un programme qui permet de :

- saisir les coordonnées du bateau en détresse **(Xd, Yd)**.
- saisir les coordonnées de **N** autres bateaux dans deux tableaux **Tx** et **Ty** (avec $1 \leq N \leq 50$) où **Tx** contient les abscisses et **Ty** contient les ordonnées. Il est à noter que deux bateaux (y compris le bateau en détresse) ne peuvent pas avoir les mêmes coordonnées.
- remplir un tableau **Td** par les distances qui séparent les différents bateaux du bateau en détresse.
- afficher les coordonnées **(X, Y)** des bateaux, du plus proche au plus loin du bateau en détresse.

N.B. : La distance **dAB** qui sépare deux points **A** et **B** de coordonnées respectives **(Xa, Ya)** et **(Xb,Yb)** est calculée comme suit :

$$dAB = \sqrt{(Xb - Xa)^2 + (Yb - Ya)^2}$$

Exemple :

Pour les coordonnées du bateau en détresse **(Xd, Yd) = (500, 300)**, le nombre de bateaux **N = 5** et les deux tableaux **Tx** et **Ty** suivants :

Tx	1000	500	100.25	-350	1200
	1	2	3	4	5

Ty	-300	400	-90	75	358.14
	1	2	3	4	5

Le calcul des distances donne le tableau **Td** suivant :

Td	781.02	100	558.48	879.28	702.41
	1	2	3	4	5

Le programme affiche les coordonnées des bateaux dans l'ordre suivant :

(500, 400) (100.25, -90) (1200, 358.14) (1000, -300) (-350, 75)

Travail demandé :

1) Analyser le problème en le décomposant en modules.
2) Analyser chaque module envisagé.

Examen du baccalauréat session juin 2016

Correction du sujet théorique d'informatique

SECTIONS : Mathématiques + Sciences Expérimentales + Sciences Techniques

Exercice 1 : (3points= 0,25*3*4)

Valider chacune des propositions ci-dessous en mettant dans la case correspondante la lettre (V) si elle est correcte ou la lettre (F) dans le cas contraire.

1) L'identificateur d'une variable :

   [V] ne doit pas commencer par un chiffre.

   [F] peut contenir un espace.

   [V] peut contenir le caractère souligné (tiret bas "_").

2) L'instructionqui permet d'affecter à une variable X, une valeur aléatoire de l'intervalle [2,10] est :

   [F] X ← 2 + Aléa (10)

   [V] X ← 2 + Aléa (9)

   [F] X ← Aléa (2 + 10)

3) Une structure de données tableau peut :

   [F] contenir des éléments de types différents.

   [F] être déclarée avec une taille maximale variable.

   [V] avoir des indices de cases de type caractère.

4) Pour le typescalaire énuméré :

   [F] les valeurs énumérées peuvent appartenir à un type prédéfini.

   [V] une valeur énumérée peutêtre affectée à une variable du même type.

   [V] les opérateurs relationnels "<", ">" et "=" peuvent être appliqués.

Exercice 2 (5 points)

1) A partir des algorithmes donnés ci-dessus, remplir la 2ème colonne du tableau suivant par un exemple de chaqueélément cité dans la 1ère colonne :**(1points = 4 * 0.25)**

Elément	Exemple
Expression booléenne	**(C = "") ou (E ≠ 0)**
	E=0
	C = ""
	E ≠ 0
Procédure prédéfinie	**Valeur(C[1],x,e)**
	Efface(C,1,1)
Paramètre formel	**C**
Paramètre effectif	**A**

Remarque : On accepte aussi **FN inconnue(A)**comme un exemple d'expression booléenne

2) Compléter l'entête de la fonction **Inconnue** par les types appropriés :**(0,5 point= 0.25*2)**

Def Fn **Inconnue** (C : ***Chaîne***) : ***Booléen***

3) Compléter le tableau de déclaration ci-dessous par les types des objets locaux de la fonction **Inconnue. (0,5 point = 0,25*2)**

Objet	*Type / Nature*
X	***Entier / Réel***
E	***Entier***

4) Parmi les variables **A**,**C**, **X** et **E**, réécrire dans le tableau ci-dessous celles qui ne sont pas visibles par le programme principal.**(0.5 point)**

(-0.25 par erreur)

Variables non visibles par le programme principal
C, X, E

5) Donner le résultat affichépar le programme **Exercice** pour chacune des valeurs de **A** suivantes : **(2 points = 4 * 0.5)**

- A = "523" **→523 Vérifie la propriété.**
- A = "-523" **→-523 Ne vérifie pas la propriété.**
- A = "5,23" **→5,23 Ne vérifie pas la propriété.**
- A = "A5B3" **→A5B3 Ne vérifie pas la propriété.**

6) En déduire le rôle de la fonction **Inconnue. (0.5 point)**

La fonction Inconnue permet de **vérifier si une chaîne est formée** uniquement **par des chiffres.**

Problème : (12 points)

1°) Analyse du programme principal

Résultat = PROC Affiche (N,Tx,Ty)
(Tx,Ty)= PROC Tri (N,Tx,Ty,Td)
Td=PROC Distance (N,Xd,Yd,Tx,Ty,Td)
(N,Xd,Yd,Tx,Ty)=PROC Saisie (N,Xd,Yd,Tx,Ty)

Tableau de déclaration des nouveaux types

Type
Tab = tableau de 50 réels

Tableau de déclaration des objets globaux

Objet	**Type / Nature**	**Rôle**
N	Entier	Nombre de bateaux les plus proches
Xd,Yd	Réel	Coordonnées du bateau en détresse
Tx	Tab	Tableau des abscisses des bateaux
Ty	Tab	Tableau des ordonnées des bateaux
Td	Tab	Tableau des distances

Affiche	Procédure	Procédure qui permet de faire l'affichage
Tri	Procédure	Procédure qui permet de faire le tri de Tx, Ty et Td
Distance	Procédure	Procédure qui permet de remplir le tableau des distances Td
Saisie	Procédure	Procédure qui permet de faire la saisie de N, Xd,Yd, Tx et Ty

2°) Analyses des modules envisagés

Analyse de la procédure Saisie

```
DEF PROC Saisie(Var N : entier ; VAR Xd,Yd : réel ; Var Tx,Ty : Tab)
Résultat = (Xd,Yd,N,Tx,Ty)
Xd,Yd = Donnée ("Introduire les coordonnées du bateau en détresse : ")
N=[]Répéter
        N=donnée ("Donner le nombre de bateaux les plus proches : ")
Jusqu'à N dans [1..50]
(Tx ,Ty) = [Répéter
        Tx[1] ,Ty[1] = Donnée("Introduire les coordonnées du bateau proche n° 1: ")
Jusqu'à ((Tx[1]<>Xd) OU (Ty[1]<>Yd))]
Pour i de 2 à N faire
    Répéter
        Tx[i] ,Ty[i] = Donnée("Introduire les coordonnées du bateau proche n° ",i," : ")
    Jusqu'à (Non (FNExiste(Tx[i],Ty[i],Tx,Ty,i)) ) ET((Tx[i]<>Xd) OU (Ty[i]<>Yd))
    Fin Pour
Fin Saisie
```

Tableau de déclaration des objets locauxde la procédure Saisie

Objet	Type / Nature	Rôle
i	Entier	Compteur
Existe	Fonction	Fonction qui permet de vérifier l'unicité des coordonnées.

Analyse de la fonction Existe:

```
DEF  FnExiste (x, y: réel ; T1,T2 : tab ; p : entier): booléen
Résultat=   existe ← test
test =[i←0]    Répéter
                    i←i+1
                    test←(T1[i]=x) et  (T2[i]=y)
               Jusqu'à  (i=p-1) ou (test=Vrai)
Fin Existe
```

Tableau de déclaration des objets locaux

Objet	T/N
i	Entier
test	booléen

Analyse de la procédure Distance

```
DEF PROC Distance (N : entier ; Xd,Yd : réel ; Tx,Ty : Tab ; Var Td : Tab)
Résultat = Td
Td= [] Pour i de 1 à N Faire
        Td[i] ←Racine_carré (carrée (Tx[i]-Xd) + carrée (Ty[i]-Yd))
    Fin Pour
Fin Distance
```

Tableau de déclaration des objets locaux de la procédure Distance

Objet	Type / Nature	Rôle
i	Entier	Compteur

Analyse de la procédure Tri

```
DEF PROC Tri (N :entier ; VarTx,Ty,Td : Tab)
Résultat = (Tx,Ty,Td)
(Tx,Ty,Td)=[] Répéter
                    Test ←Faux
                    Pour i de 1 à N-1 Faire
                        Si Td[i] >Td[i+1] Alors
                                        PROC Permut (Td[i],Td[i+1])
                                        PROC Permut (Tx[i],Tx[i+1])
                                        PROC Permut (Ty[i],Ty[i+1])
                                        Test ←Vrai
                        Fin Si
FinPour
                    N← N-1
              Jusqu'à (Test = Faux) OU (N=1)
Fin Tri
```

Tableau de déclaration des objets locaux de la procédure Tri

Objet	Type / Nature	Rôle
i	Entier	Compteur
Test	Booléen	Tester s'il y a eu une permutation ou non
Permut	Procédure	Procédure qui permet de faire la permutation de deux cases.

Analyse de la procédure Permut

```
DEF PROC Permut(Varx,y : réel)
Résultat = (x,y)
(x,y) = aux ← x
        x ← y
        y← aux
Fin Permut
```

Tableau de déclaration des objets locaux de la procédure Permut

Objet	Type / Nature	Rôle
aux	réel	Variable auxiliaire

Analyse de la procédure Affiche

```
DEF PROC Affiche (N : entier ; Tx,Ty : Tab)
Résultat = [ ]  Pour i de 1 à N faire
                    Ecrire ("(",Tx[i]," , " , Ty[i] , ") "    )
              Fin Pour
Fin Affiche
```

Tableau de déclaration des objets de la procédure Affiche

Objet	Type / Nature	Rôle
i	Entier	Compteur

Barème : 12 points

Toute solution équivalente sera acceptée
-0.25 par type d'erreur

Action	Nombre de points
Programme principal : • Modularité • Cohérence (mode de passage, conformité entre nombre, ordre et type des paramètres)	**2 points :** • 1 • 1 = 0.25*4
Saisie de n: • Lecture • Contrôle	**0.5 point :** • 0.25 • 0.25
Saisie de Xd et Yd:	**0.5 point** = 0.25+0.25

Saisie de Tx et Ty:
• Initialisation (lecture du premier élément)
• Parcours
• lecture d'un élément de Tx et un élément de Ty
• Contrôle de saisie
  ○ Testd'existence + Traitement de Existe
  ○ Test de différence avec Xd et Yd
	2.5 points :
	• 0.25
	• 0.5
	• 0.25+0.25
	• 0.25+0.75
	• 0.25

Remplissage de Td: • parcours • calcul	**1 point :** • 0.5 • 0.5

Tri :
• Parcours
• Comparaison
• permutation
  ○ dans Tx
  ○ dans Ty
  ○ dans Td
	3 points :
	• 1= 0.5 + 0.5
	• 0.5
	• 0.5
	• 0.5
	• 0.5

Affichage : • Parcours • Ecriture	**1.5 points** • 0.5 • 1= 0.5+0.5
TDNT **TDOG** **TDOL**	**0.25 point** **0. 5 point** **0.25 point**

RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ÉDUCATION ◈◈◈◈ **EXAMEN DU BACCALAURÉAT** SESSION **2015**	Épreuve : **Informatique**
	Durée : **1 H 30 mn**
	Coefficient : **0,5**
Sections : **Mathématiques, Sciences expérimentales et Sciences Techniques**	

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

Nom et prénom : .....

Date et lieu de naissance : .....

Signatures des surveillants

.....

.....

Le sujet comporte 4 pages numérotées de 1/4 à 4/4.
Les réponses à l'exercice 1 et 2 doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve

Exercice 1 (3,75 points)

Afin de réaliser les tâches décrites dans la première colonne du tableau suivant, un élève fournit les propositions suivantes. Remplir la colonne "**Correction**" en apportant les corrections nécessaires pour que ces propositions soient les plus adéquates relativement au choix de la structure itérative.

Tâche	Proposition	Correction
Saisir un entier positif **n**	n= [n= donnée ("Saisir un entier positif :")] Tant que (n<0) faire n= donnée ("Saisir un entier positif :") Fin tant que	....
Chercher la valeur maximale dans un tableau **T** de taille **n**.	Max = [i ← 1, Max ← T[1]] Répéter [ ] Si (T [i]> Max) alors Max ←T[i] Fin Si i←i+1 Jusqu'à (i > n)	....
Vérifier l'existence d'un caractère **C** dans un tableau **T** de **n** caractères.	Trouve = [Trouve ←faux] Pour i de 1 à n faire [ ]Si (T[i] = C) Alors Trouve ←Vrai Fin si Fin pour	....

Ne rien écrire ici

Exercice 2 (5,25 points)

Soit **U0** un entier naturel de quatre chiffres. A l'aide de ses quatre chiffres, on compose le plus grand entier et le plus petit entier formés par ces chiffres.

La différence de ces deux nombres donne **U1,** qui sera soumis au même traitement pour donner **U2**, etc. Jusqu'à ce que la suite **U** devienne **stationnaire,** c'est-à-dire, à un certain terme elle devient constante (ne change plus de valeur).

Soit l'algorithme suivant nommé **Suite** et permettant de déterminer les termes d'une suite **U** ayant comme premier terme **U0**, de les ranger dans un tableau **T** et de l'afficher (avec **Max** et **Min** sont deux modules qui déterminent respectivement le plus grand entier et le plus petit entier formés à partir des chiffres de **Ui** avec **i** >0).

```
0- Début Suite
1- Répéter
      Lire (U0)
   Jusqu'à (U0 ≥ 1000) et (U0 ≤ 9999)
2- i ←1
   T[1]←U0
   Répéter
      i←i+1
      T[i] ← FN Max(U0) – FN Min(U0)
      U0← T[i]
   Jusqu'à (T[i] = T[i-1])
3- Proc Afficher (T, i)
4- Fin Suite
```

Travail demandé :

Pour chacune des questions suivantes, cocher la ou les bonnes réponses.

1- Par quel appel peut-on remplacer la séquence **1** de l'algorithme **Suite** ?

- ☐ Proc Saisir (N)
- ☐ Procédure Saisir (Var N : entier)
- ☐ Proc Saisir (U0)
- ☐ U0← Proc saisir (N)

2- Quels sont les en-têtes qui correspondent à la déclaration de la procédure **Afficher** ?

- ☐ DEF Proc Afficher (Var T : tab)
- ☐ DEF Proc Afficher (T : tab ; N : entier)
- ☐ DEF Proc Afficher (i : entier ; T : tab)
- ☐ DEF Proc Afficher (T[i] : entier)

3- L'en-tête suivant de la fonction **Max** est erroné : **DEF FN Max (X : entier)**
Quel est l'origine de l'erreur ?

- ☐ Le mode de passage des paramètres est erroné.
- ☐ Le nom du paramètre effectif est différent du nom du paramètre formel.
- ☐ Le type du résultat est manquant.
- ☐ Le type du paramètre effectif est incompatible avec celui du paramètre formel.

Ne rien écrire ici

4- Si on veut remplacer la séquence 2 par l'appel d'un module :

a. Quelle sera sa nature ?

☐ Une procédure ☐ Une fonction

b. Quels seront les paramètres effectifs à utiliser ?

☐ T, i et U0
☐ T[i] et U0
☐ T et U0
☐ T et i

5- Quel sera le tableau de déclaration des objets de l'algorithme **Suite** ?

☐ **T.D.O.G**

Objet	**Type**
T	Tab
U0	Entier

☐ **T.D.O.G**

Objet	**Type**
T	Tab
I, U0	Entier
Max, Min	Fonction
Afficher	Procédure

6- Pour **U0** égale à **5360**, quel sera le résultat de l'affichage de l'algorithme **Suite** ?

☐ **T**

5843	5085	7992	7173	6354	3087	8352	6147	6174

☐ **T**

5843	5085	2970	6930	5940	4950	4950

Ne rien écrire ici

Problème (11 points)

Un nombre **M** est dit « **nombre premier sûr** », s'il est un nombre premier de la forme **2*p + 1** avec **p** un nombre premier.

Exemples **:**

- ✓ Si **M** = 11, alors **M** est un nombre premier sûr. En effet, **11** est premier et il peut s'écrire sous la forme **2*p+1** où **p = 5** qui est un nombre premier.
- ✓ Si **M** = 31, alors **M** n'est pas un nombre premier sûr. En effet, **31** est premier et il peut s'écrire sous la forme **2*p+1** où **p = 15** qui n'est pas un nombre premier.

NB **:** Un nombre entier supérieur à **1** est dit premier s'il n'est divisible que par **1** et par lui-même.

On se propose d'écrire un programme qui permet de :

1. Remplir un tableau **T** par **N** entiers strictement supérieurs à 1 (avec $10 \leq N < 45$).
2. Trier dans l'ordre croissant les éléments premiers sûrs du tableau **T** suivis du reste des éléments sans tri.
3. Afficher le tableau **T** résultant.

Exemple : Pour **N** = 10 et le tableau **T** suivant :

T	5	25	59	23	13	47	31	100	7	107
	1	2	3	4	5	6	7	8	9	10

Le programme affichera le contenu du tableau suivant :

T	5	7	23	47	59	107	25	13	31	100
	1	2	3	4	5	6	7	8	9	10

Eléments premiers sûrs triés dans un ordre croissant

Eléments non premiers sûrs

Travail demandé :

1) Analyser le problème en le décomposant en modules.

2) Analyser chacun des modules envisagés.

Bac 2015
Corrigé : Informatique
Sections : Math, Techniques et Sciences Expérimentales

Exercice 1 (1.25 x 3 = 3.75 points)

Afin de réaliser les tâches décrites dans la première colonne du tableau suivant, un élève fournit les propositions suivantes. Apporter les corrections nécessaires pour que ces propositions soient les plus adéquates relativement au choix de la structure itérative.

Tâche	Proposition	Correction
Saisir un entier **n** positif.	n= [n= donnée ("Saisir un entier positif :")] Tant que (n<0) faire n= donnée ("Saisir un entier positif :") Fin tant que	**n=[ ]** **Répéter** **n= donnée ("Saisir un entier positif :")** **Jusqu'à (n ≥ 0)**
Chercher la valeur maximale dans un tableau **T** de taille **n**.	Max = [i ← 1, Max← t[1]] Répéter [ ] Si (T [i]> Max) alors Max ←T[i] Fin Si i←i+1 Jusqu'à (i > n)	**Max = [Max← T[1]]** **Pour i de 2 à n faire** **Si (T[i]> Max) alors** **Max ←T[i]** **Fin Si** **Fin pour**
Vérifier l'existence d'un caractère **C** dans un tableau **T** de **n** caractères.	Trouve= [Trouve ←faux] Pour i de 1 à n faire [ ] Si (T[i] = C) Alors Trouve ←Vrai Fin si Fin pour	**Trouve= [i ← 0]** **Répéter** **i←i+1** **Trouve ← T[i] = C** **Jusqu'à (Trouve) ou (i= n)**

Exercice 2 (0.75 x 7 = 5.25 points)

Pour chacune des questions suivantes, cochez la ou les bonnes réponses.

1- Par quel appel peut-on remplacer la séquence 1 de cet algorithme ?

- [ ] Proc Saisir (N)
- [ ] Procedure Saisir (Var N : entier)
- [x] Proc Saisir (U0)
- [ ] U0← Proc saisir (N)

2- Quelles sont les entêtes qui correspondent à la procédure **Afficher**?

- [ ] DEF Proc Afficher (Var T : tab)
- [x] DEF Proc Afficher (T : tab ; N : entier)
- [ ] DEF Proc Afficher (i : entier ; T : tab)
- [ ] DEF Proc Afficher (T[i] : entier)

3- L'entête suivante de la fonction Max est erronée : DEF FN **Max** (X : entier)
Quel est l'origine de l'erreur ?

- [ ] Le mode de passage des paramètres est erroné.
- [ ] Le nom du paramètre effectif est différent du nom du paramètre formel.
- [x] Le type du résultat manque.
- [ ] Le type du paramètre effectif est incompatible avec celui du paramètre formel.

4- Si on veut remplacer la séquence 2 par l'appel d'un module :

a. Quelle sera sa nature ?

- [x] Procédure
- [ ] Fonction

b. Quels seront les paramètres à utiliser ?

- [x] T, i et U0
- [ ] T[i] et U0
- [ ] T et U0
- [ ] T et i

5- Quel sera le tableau de déclaration des objets de l'algorithme Suite ?

- [ ]

Objet	Type
T	Tab
U0	Entier

- [x]

Objet	Type
T	Tab
I, U0	Entier
Max, Min	Fonction
Afficher	Procédure

6- Pour une valeur donnée d'**U0** égale à **5360.** Quel sera le résultat de l'affichage de l'algorithme **Suite** ?

- [x] **T**

5843	5085	7992	7173	6354	3087	8352	6147	6174

- [ ] **T**

5843	5085	2970	6930	5940	4950	4950

Problème (11 points)

TDNT

Type
Vect = tableau de 44 entiers

a) ANALYSE DU PROGRAMME PRINCIPAL

Résultat = PROC **afficher** (T, n)
T = PROC **trier** (T, nb)
T, nb = PROC **ranger** (T, n, nb)
T, n = PROC **remplir** (T, n)

TDOG

Objet	T/N
T	Vect
n, nb	Entier
Remplir	Procédure
Ranger	Procédure
Trier	Procédure
Afficher	Procédure

b) ANALYSE DE LA PROCEDURE REMPLIR

```
DEF PROC remplir (var t : vect ; var n : entier)
Résultat = t, n
t= [ ]
Pour i de 1 à n faire
    Répéter
      t[i]= donnée ("saisir un entier :")
    Jusqu'à (t [i]>1)
Fin pour
n= [ ]
Répéter
  n= donnée ("saisir un entier :")
Jusqu'à (n dans [10..44])
```

Objet	Type
i	Entier

c) ANALYSE DE LA PROCEDURE RANGER

```
DEF PROC Ranger (Var t : vect ; n : entier ; Var nb: entier)
Résultat = t, nb
    (t, nb) = [nb←0]
Pour i de 1 à n faire
  Si (FN Premier (t[i]) et (FN Premier ((t[i]-1) div 2) alors
        nb← nb+1
        aux← t[i]
        t[i] ← t[nb]
        t[nb] ← aux
   Fin si
Fin pour
Fin Ranger
```

Objet	Type
Premier	Fonction
PremSur	Fonction
i, j	Entier
aux	Entier

d) ANALYSE DE LA FONCTION

DEF FN Premier (r : entier) : booléen
Résultat = premier ← (r mod d=0)
d = [d ←1]
Répéter
d←d+1
Jusqu'à (r mod d =0) ou (d > racinecarré(r))
Fin premier

Objet	Type
d	Entier

e) ANALYSE DE LA PROCEDURE TRIER

Def Proc TRIER (VAR T : VECT ; N : ENTIER)

Résultat = T
T = [ ] Répéter
Echange ← faux
Pour i de 1 à n-1 faire
Si (T[i] > T[i+1]) alors
Aux←T[i]
T[i]←T[i+1]
T[i+1]←Aux
Echange ← vrai
Fin si
Fin pour
n←n-1
Jusqu'à (n=1) ou (Echange = faux)

Objet	Type
Aux	Entier
i	Entier
Echange	booléen

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ◈◇◈ **EXAMEN DU BACCALAUREAT** SESSION DE JUIN **2014**	Epreuve : **INFORMATIQUE** Durée : **1 h 30** Coefficient : **0,5**
Sections : Mathématiques, Sciences Expérimentales et Sciences Techniques	

Le sujet comporte 04 pages.

NB. Les réponses aux EXERCICES doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie qui contiendra les réponses au PROBLEME.

Exercice 1 : (3.5 points)

Soient les tableaux de déclarations suivants :

Tableau de déclaration des nouveaux types

Types
Jour_semaine = (Dimanche, Lundi, Mardi, Mercredi, Jeudi, Vendredi, Samedi)
Jour_travail = Lundi ..Vendredi
Phrase = chaîne [10]
Tab = tableau [Lundi..Vendredi] de Phrase

Tableau de déclaration des objets

Objets	Type / Nature
a	Entier
T	Tab
Ch	Phrase
i, j	Jour_travail

Compléter le tableau suivant en mettant **"Vrai"** si l'instruction est valide ou **"Faux"** dans le cas contraire. Justifier la réponse en cas d'instruction jugée invalide.

Instruction	Vrai/ faux	Justification
a ← ORD(SUCC(Vendredi))	.....	....
T[Vendredi] ← "Bonjour"	.....	....
Ch[15] ← Sous-chaîne ("Bac",1,1)	.....	....
Lire (j)	.....	....
Pour i de Lundi à Vendredi faire Ecrire (T[i]) Fin pour	.....	....

Exercice 2 : (4.5 points)

En classe, un enseignant a donné un exercice permettant de vérifier si un triplet de réels (a, b, c) est dit triplet de Pythagore c'est-à-dire $a^2 + b^2 = c^2$. Un élève propose une solution formée par les deux algorithmes suivants :

L'algorithme du programme appelant :

```
0) Début Prg_Appelant
1) Lire (n1)
2) Lire (n2)
3) Lire (n3)
4) Si  R=Vrai alors
     Ecrire (n1,",", n2,"et", n3, "forment un triplet de Pythagore")
   FinSi
5) Fin Prg_Appelant
```

Et l'algorithme, sans en-tête, de la fonction **Pythagore** :

```
0)
1) Si carré(a) + carré(b) = carré(c) alors
       Pythagore← Vrai
   Sinon
       Pythagore← Faux
   FinSi

2) Fin Pythagore
```

En passant à la correction de la proposition de l'élève, l'enseignant vous demande de répondre aux questions suivantes :

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

En-tête proposée pour la définition de la fonction Pythagore	**Réponse**
DEF FN Pythagore (a,b,c : Entier) : booléen	
DEF FN Pythagore (n1,n2,n3 : réel) : booléen	
DEF FN Pythagore (a,b,c : Réel) : booléen	
DEF FN Pythagore (a,b,c : Réel) : réel	

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.

.....

.....

.....

.....

.....

.....

.....

.....

.....

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

.....

.....

Problème : (12 points)

On se propose de crypter un message composé par des mots séparés par un seul espace et ne contenant aucun signe de ponctuation (, ; . : ! ?) en utilisant le principe suivant :

1) Placer chaque mot du message initial dans une case d'un tableau **T**. On suppose que le message est composé d'au maximum 20 mots.
2) Pour chaque élément du tableau **T**, ajouter autant de fois le caractère "*" pour que sa longueur sera égale à celle du mot le plus long dans le tableau **T**.
3) Dans un nouveau tableau **T1** de taille **N1** (**N1**=longeur du mot le plus long), répartir les lettres du mot se trouvant dans la case **T[1]** de façon à placer la lettre d'indice **i** du mot dans la case d'indice **i** du tableau **T1**.
4) Répartir de la même façon les lettres du mot contenu dans la case **T[2]** en concaténant à chaque fois la lettre d'indice **i** avec le contenu de la case **i** du tableau **T1**
5) Répartir de la même façon le reste des mots de **T** dans **T1**.
6) Concaténer les mots obtenus dans **T1** en les séparant par un espace pour obtenir le message crypté.

Exemple : *Si le message à crypter est "Bonjour Sami j'ai fini mon travail", les étapes de cryptage sont :*

Etape 1 : *Répartir les mots du message dans le tableau T :*

T=	Bonjour	Sami	j'ai	fini	mon	travail

Etape 2 : *Ajouter le caractère "*" autant de fois pour obtenir des mots dont la longueur de chacun est égale à celle du mot le plus long.*

Etant donné que "Bonjour" est le mot le plus long du message (7 caractères), on obtient le tableau **T** suivant :

T=	Bonjour	Sami***	j'ai***	fini***	mon****	travail

Etape 3 : *Répartir les lettres de T[1] dans T1*

T=	B	o	n	j	o	u	r

Etape 4 : *Répartir les lettres de T[2] dans T1*

T1 =	BS	oa	nm	ji	o*	u*	r*

Etapes suivantes : *Répartir le reste des mots de T dans T1*

T1 =	BSjfmt	oa'ior	nmanna	jiii*v	o****a	u****i	r****l

Le message crypté sera alors "BSjfmt oa'ior nmanna jiii*v o****a u****i r****l"

Travail demandé :

1. Analyser le problème en le décomposant en modules.
2. Analyser chacun des modules proposés.

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 ← ORD(SUCC(Vendredi))	Vrai	
T[Vendredi] ← ″Bonjour″	Vrai	
Ch[15] ← sous-chaîne ("Bac",1,1)	Faux	Il est impossible d'affecter une chaîne à un caractère. On acceptera aussi 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 Ecrire (T[i]) 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 ← 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 carre(a) + carre(b) = 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	Chaine	Chaine à cryptér
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 :chaine)
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 : chaine) : 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 : chaine) : chaine
Résultat = Cypter ← Ch
Ch = [Ch←""] **Pour i de 1 à lmax faire**
Ch←Ch + T1[i]+ "␣"
Fin pour
Ch←sous-chaine(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

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ◈◈◈ **EXAMEN DU BACCALAUREAT** **SESSION DE JUIN 2013**	Epreuve : **Informatique** Durée : **1 h 30** Coefficient : **0,5**
Sections : **Mathématiques, Sciences Expérimentales et Sciences Techniques**	

NB. Les réponses aux EXERCICES doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie qui contiendra les réponses au PROBLEME.

****

Exercice 1 : (3 points)

Pour chacune des instructions suivantes, valider chaque proposition en mettant dans la case correspondante la lettre **V** si elle est correcte ou **F** dans le cas contraire.

a. Soit l'instruction **C ← Sous_chaîne ("Baccalauréat",4,1)**.

Elle permet d'affecter le caractère **"c"** à la variable **C**. ☐

La variable **C** doit être déclarée de type **caractère**. ☐

La variable **C** doit être déclarée de type **Chaîne**. ☐

b. L'instruction **X ← Aléatoire (6) + 4** permet d'affecter à la variable **X** une valeur aléatoire de l'intervalle

[4,6] ☐

[4,10] ☐

[4,9] ☐

c. L'instruction **R ← Arrondi (12.5)** permet d'affecter à la variable **R**

l'entier **12** ☐

l'entier **13** ☐

le réel **13.0** ☐

d. Soit l'affectation suivante **C ← Majus("?")**.

Elle permet d'affecter à la variable **C** le caractère "?" en gras. ☐

Elle permet d'affecter à la variable **C** le caractère "?". ☐

La variable **C** doit être de type **Caractère**. ☐

Exercice 2 : (2 points)

Soit la partie déclarative suivante d'un programme Pascal :

```
Program Composer ;
   CONST mot1 = 'informatique' ; mot2 = '3D' ;
   VAR mot3, mot4 : string ; n, m : integer ;
```

En utilisant des fonctions et des procédures prédéfinies, donner **les instructions Pascal** permettant de réaliser les traitements suivants :

a- A partir de la constante **mot1**, mettre dans la variable **mot3** le terme "format".

.....

b- A partir de **mot3** et **mot2**, mettre dans la variable **mot4** le terme "format 3D".

.....

c- Mettre dans **n** la longueur de la chaîne **mot4**.

.....

d- A partir de la constante **mot2**, mettre dans **m** la valeur **3**.

.....

Exercice 3 : (3 Points)

Soit le type **Examen** contenant les valeurs suivantes :

Math, Anglais, Physique et **Informatique**

1. Qu'appelle-t-on le type **Examen** décrit ci-dessus ?

   .....

2. Proposer une déclaration Pascal du type **Examen** en respectant l'ordre des valeurs proposé ci-dessus.

   .....

   .....

3. Compléter le tableau ci-dessous par les types et les valeurs des variables **A**, **B** et **C** après exécution des instructions Pascal suivantes :

```
A := PRED (Informatique) ;
B := ORD (Anglais) * 8 DIV 4 ;
C := (Math < Physique) ;
```

Variable	Type	Valeur
A	.....	.....
B	.....	.....
C	.....	.....

Problème : (12 points)

Soit **T** un tableau de **N** entiers **(avec 6≤N≤50)**. On se propose de trier le tableau **T** dans l'ordre croissant en utilisant le principe suivant :

1. On parcourt le tableau **T** de gauche à droite en comparant les éléments de **T** deux à deux **(T[i] avec T[i+1])** et en les permutant si nécessaire. Le premier parcours permet de placer le plus grand élément dans la dernière case.
2. On parcourt le tableau de droite à gauche (sans tenir compte de la dernière case : case triée) tout en comparant chaque deux éléments consécutifs de **T** et en les permutant si nécessaire pour placer le plus petit élément à sa bonne place (**case n°1**).
3. On refait les étapes **1** et **2** en parcourant le tableau tantôt de gauche à droite et tantôt de droite à gauche sans tenir compte des cases triées. Le traitement sera arrêté lorsque le tableau est trié.

Exemple

Soient **N=6** et le tableau **T** suivant :

13	6	4	20	5	9
1	2	3	4	5	6

En appliquant le principe décrit ci-dessus sur le tableau **T** précédent, on obtient les étapes suivantes :

1. Le 1er parcours de **gauche à droite**, permet de placer la valeur **20** (plus grand élément de **T**) dans la case **n°6**.

6	4	13	5	9	**20**
1	2	3	4	5	6

2. Le 1er parcours de **droite à gauche** (sans tenir compte de **T[6]**), permet de placer la valeur **4** (plus petit élément de **T**) dans la case **n°1**.

4	6	5	13	9	**20**
1	2	3	4	5	6

3. Le 2ème parcours de **gauche à droite** (sans tenir compte de **T[1]** et **T[6]**) permet de placer la valeur **13** (plus grand élément de la portion du tableau allant de la case **2** à la case **5**) dans la case **n°5**.

4	5	6	9	**13**	**20**
1	2	3	4	5	6

4. Durant le 2ème parcours de **droite à gauche** (de la case **4** à la case **2**), aucune permutation n'a été faite, donc le tableau est trié.

4	**5**	**6**	**9**	**13**	**20**
1	2	3	4	5	6

On se propose d'écrire un programme qui permet de remplir un tableau **T** par **N** entiers puis de trier **T** selon le principe décrit précédemment et d'afficher le tableau trié.

Questions

1. Analyser le problème en le décomposant en modules.
2. Analyser les modules envisagés.

EXAMEN DU BACCALAUREAT 2013

SECTION : Math + Sc. Exp. + Sc. Tech. **EPREUVE : Informatique**

CORRIGE

Exercice 1 : (3 points= 4 * 3 * 0.25)

Pour chacune des instructions Pascal suivantes, valider chaque proposition en mettant dans la case correspondante la lettre **V** si elle est correcte ou **F** dans le cas contraire.

a. C ← Sous_chaîne ("Baccalauréat",4,1)

Elle permet d'affecter le caractère **"c"** à la variable **C.**	**V**
La variable **C** doit être déclarée de type **caractère.**	**F**
La variable **C** doit être déclarée de type **chaîne.**	**V**

b. X ← Aléatoire (6) + 4

Permet d'affecter à la variable **X** une valeur aléatoire de l'intervalle [4,6]	**F**
Permet d'affecter à la variable **X** une valeur aléatoire de l'intervalle [4,10]	**F**
Permet d'affecter à la variable **X** une valeur aléatoire de l'intervalle [4,9]	**V**

c. R ← Arrondi (12.5)

Permet d'affecter à la variable **R** l'entier **12**	**F**
Permet d'affecter à la variable **R** l'entier **13**	**V**
Permet d'affecter à la variable **R** le réel **13.0**	**F**

d. C ← Majus("?")

Permet d'affecter à la variable **C** le caractère '?' en gras	**F**
Permet d'affecter à la variable **C** le caractère '?'	**V**
La variable **C** doit être de type **Caractère**	**V**

Exercice 2:(2 points = 4*0.5)

Soit la partie déclarative suivante d'un programme Pascal :

Program Composer ;
CONST mot1 = 'informatique' ; mot2 = '3D' ;
VAR mot3, mot4 : string ; n, m : integer ;

En utilisant **des fonctions et des procédures prédéfinies**, donner **les instructions Pascal** permettant de réaliser les traitements suivants :

a- A partir de la constante **mot1**, mettre dans la variable **mot3** le terme "format".

mot3 :=copy (mot1,3,6) ;

b- A partir de **mot3** et **mot2**, mettre dans la variable **mot4** le terme "format 3D".

mot4 := concat (mot3 , ' ' , mot2) ;

ou **mot4 := mot3+mot2 ;**

c- Mettre dans **n** la longueur de la chaîne **mot4**

n := length (mot4) ;

d- A partir de la constante **mot2**, mettre dans **m** la valeur **3**.

Val (copy(mot2,1,1),m,n) ;

Exercice 3 : (3 Points= 0.75+0.75+0.25*6)

Soit le type **Examen** contenant les valeurs suivantes :
Math, Anglais, Physique et **Informatique**

1. Qu'appelle-t-on le type **Examen** décrit ci-dessus ?

Type scalaire énuméré

2. Proposer une déclaration Pascal du type **Examen** en respectant l'ordre des valeurs proposé ci-dessus.

Examen= (**Math, Anglais, Physique, Informatique) ;**

3. Compléter le tableau ci-dessous par les types et les valeurs des variables **A**, **B** et **C** après exécution des instructions suivantes :

A := PRED (Informatique) ;

B := ORD (Anglais) * 8 DIV 4 ;

C := (Math< Physique) ;

Variable	Type	Valeur
A	Examen	Physique
B	Tout type numérique	2
C	Boolean /Booléen	True

Problème : (12 points)

Analyse du programme principal :

Résultat = Proc affiche(T_f,n)
T_f = Proc Tri (T_i,n)
T_i = Proc Remplissage (T_i,n)
N = Proc saisie(n)

NB : T_f représente l'état final du tableau T
T_i représente l'état initial du tableau T

T.D.N.T

Type
Tab = tableau de 50 entiers

T.D.O.G

Objet	Type /Nature	Rôle
T	Tab	Tableau à trier
N	Entier	Nombre d'éléments du tableau
Saisie	Procedure	Permet de saisir le nombre d'éléments du tableau T
Remplissage	Procedure	Permet de remplir le tableau T
Tri	Procedure	Permet de trier le tableau T
affiche	Procedure	Permet d'afficher le tableau T après tri

Analyse de la procédure saisie

DEF PROC saisie (var n : entier)

Résultat= n

n=[ ]répéter

n= donnée("saisir le nombre d'entiers : ")

jusqu'à (n dans [6..50])

Fin saisie

Analyse de la procédure remplissage

DEF PROC remplissage (var T : tab ; n : entier)

Résultat= T

T=[ ]Pour i de 1 à n faire

T [i]= donnée (" Donner T[", i, "] : ")

FinPour

Fin remplir

T.D.O.L

Objet	T/N	Rôle
i	Entier	Compteur

Analyse de la procédure tri

DEF PROC tri (var T: tab; n: entier)

Résultat= T

T=[j←0] Répéter

Permut←faux

j← j+1

Pour i de j à n-1 faire

Si (T[i]>T[i+1]) alors

Permut← vrai

aux←T[i]

T[i]←T[i+1]

T[i+1]← aux

FinSi

FinPour

Si (permut = vrai) Alors

Permut←faux

Pour i de n-1 à j+1 faire

Si (T[i]<T[i-1]) alors

Permut← vrai

Aux←T[i]

T[i]←T[i-1]

T[i-1]← aux

FinSi

FinPour

FinSi

n←n-1

Jusqu'à (permut = faux) ou (j ≥ n)

Fin trier

T.D.O.L

Objet	T/N	Rôle
i , j	Entier	Compteur
aux	Entier	Variable auxiliaire
permut	booléen	Test de permutation

Analyse de la procédure affiche

DEF PROC affiche (T : tab ; n : entier)

Résultat= [ ]Pour i de 1 à n faire

Ecrire(T [i])

FinPour

Fin affiche

T.D.O.L

Objet	**T/N**	**Rôle**
I	Entier	Compteur

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ◈◈◈	EXAMEN DU BACCALAUREAT SESSION DE JUIN 2012		
	Epreuve : Informatique	Durée : 1h30	Cœfficient : 0,5
SECTIONS : Mathématiques + Sciences Expérimentales + Sciences Techniques		SESSION PRINCIPALE	

N.B : La réponse à l'EXERCICE 1 doit être rédigée sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie qui contiendra les réponses à l'EXERCICE 2 et à la PARTIE II

****

Partie I : 6 points

Exercice 1: (2 points)

Compléter le tableau suivant par les valeurs des variables indiquées sachant que toutes les instructions sont correctes.

Instructions	Valeurs
X ← Tronc (11.8) Y← Arrondi (11.8)	X= ..... Y=.....
Valeur ("138.25" , N , E)	N= ..... E=.....
Convch (138.25 , Ch)	Ch=.....
Ch1←"information" Efface (ch1 , 3 , 6)	Ch1=.....
Ch1←"information" Ch2← sous_chaine (ch1 , 3 , 6)	Ch1=..... Ch2=.....

Exercice 2 : (4 points)

1) Ecrire un algorithme d'une fonction **FACT** permettant de calculer la factorielle d'un entier naturel **N.** On rappelle que la factorielle de **N** est **N! = 1 * 2 * … * N**
2) Utiliser la fonction **FACT** pour écrire l'algorithme d'une fonction **SOMME** qui permet de calculer la somme **S** suivante avec **N** un entier impair :

S= 1+ 1/3! + 1/5! + … + 1/N!

Partie 2 : (14 points)

On se propose d'écrire un programme qui saisit un entier naturel **N (2 ≤ N ≤ 20)**, puis remplit un tableau **T** par **N** nombres complexes de la forme **a+bi** avec **a** et **b** deux entiers naturels non nuls. Chaque suite d'éléments du tableau T qui ont le même module sera affiché sur une ligne à part. On rappelle que le module d'un nombre complexe de la forme **a+bi** est $\sqrt{a^2+b^2}$.

Pour réaliser le traitement demandé on suivra les étapes suivantes :

- Remplir un tableau **M** par les modules des éléments de **T** de façon à ce que **M[i]** soit le module du nombre complexe **T[i]**.
- Trier simultanément les deux tableaux **T** et **M** selon l'ordre décroissant des valeurs du tableau M.
- Afficher chaque suite d'éléments du tableau T qui ont le même module sur une ligne à part.

Exemple :

Pour **N=6** et pour le tableau **T** suivant :

T	2+3i	2+15i	2+17i	23+3i	17+2i	15+2i
	1	2	3	4	5	6

- Le remplissage de **M** donne le tableau suivant :

M	3. 60	15.13	17.12	23.19	17.12	15.13
	1	2	3	4	5	6

- Après le tri on obtient les deux tableaux suivants :

T	23+3i	2+17i	17+2i	2+15i	15+2i	2+3i
	1	2	3	4	5	6
M	23.19	17.12	17.12	15.13	15.13	3.60
	1	2	3	4	5	6

- Le programme affiche les lignes suivantes :

23+3i
2+17i 17+2i
2+15i 15+2i
2+3i

Travail demandé:

1. Analyser le problème en le décomposant en modules.
2. Analyser chacun des modules proposés.

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ◈◈◈	EXAMEN DU BACCALAUREAT JUIN 2012
	Corrigé de l'épreuve Informatique
Sections : Mathématiques + Sciences expérimentales + Sciences techniques	Session principale

Partie I : 6 points

Exercice n°1: (2 points = 8 * 0,25)

Donner les valeurs des variables indiquées :

Instructions	Valeurs		
X ← Tronc (11.8) Y← Arrondi (11.8)	***X= 11*** ***Y=12***		
Valeur (''138.25'',N , E)	***N= 138.25*** ***E= 0***	***ou bien***	***N= 0*** ***E= 4***
Convch (138.25,Ch)	***Ch="138.25"***		
Ch1←''information'' Efface (ch1,3,6)	***Ch1= "inion "***		
Ch1←''information'' Ch 2← sous_chaine(ch1,3,6)	***Ch1= "information"*** ***Ch2= "format"***		

Exercice 2 : (4 points)

1°) Algorithme de la fonction Fact :

```
0) Def FN Fact (n : entier) : entier
1) F← 1
   Pour i de 2 à n faire
     F ← F * i
   Fin Pour
2) Fact ← F
3) Fin fact
```

2°)Algorithme de la fonction somme :

```
0) Def FN Somme (n : entier) : réel
1) S← 1
   Pour i de 1 à (n div 2) faire
     S← S+( 1/FN Fact(2*i +1))
   Fin Pour
2) Somme←S
3) Fin Somme
```

N. B. : Le type de la fonction Fact peut être : Entier Long ou Réel.

Partie II : 14 points

1/ Analyse du programme principal

Résultat = Proc Affiche (T,M,n)
(T,M) = Proc Trier(T,M,n)
(T,M,n) = Proc Saisie(n)
Proc Lecture(T,M,n)

T.D.N.T

Type
Tab1 = tableau de 20 chaines
Tab2 = tableau de 20 réels

T.D.O.G

Objet	*Type /nature*	*Rôle*
T	*Tab1*	*Tableau contenant des nombre complexes*
M	*Tab2*	*Tableau des modules des nombres complexes*
N	*Octet*	*Nombres des éléments du tableau T.*
Affiche	*Procédure*	*Affichage de chaque suite sur une ligne à part*
Trier	*Procédure*	*Trier les éléments de T et M selon l'ordre décroissant*
Saisie	*Procédure*	*de leurs modules*
Lecture	*Procédure*	*Saisie de n*
		Remplissage de T et M

2/Analyse des modules

Analyse de la procédure saisie :

```
Def  Proc saisie (Var  n : octet)
Résultat = n
n =  [  ] Répéter
          n = donnée ("Entrer le nombre des éléments du tableau : ")
      Jusqu'à (n Dans [2..20])
Fin saisie
```

Analyse de la procédure lecture :

```
Def  Proc Lecture (Var T : tab1 ; Var M : Tab2 ; n : octet)
Résultat = (T,M)
(T,M)  =  [  ] Pour i de 1 à N faire
                Proc Saisie_partie(a)
                Proc Saisie_partie(b)
                Convch (a,ch1)
                Convch (b,ch2)
                T[i]← Ch1 + "+" + Ch2  + "i"
                M[i] ← racinecarrée (carré(a)+carré(b))
              FinPour
Fin Lecture
```

T.D.O.L

Objet	*Type*	*Rôle*
i	*Octet*	*Compteur*
a	*Entier*	*Désigne la partie réelle du nombre complexe*
b	*Entier*	*T[i]*
Ch1	*Chaine*	*Désigne la partie imaginaire du nombre*
Ch2	*Chaine*	*complexe T[i]*
Saisie_partie	*procédure*	*Conversion de a en chaine*
		Conversion de b en chaine
		Saisie d'un entier positif

Analyse de la procédure saisie_partie :

```
Def  Proc saisie_partie (Var k : entier)
Résultat = k
k = [  ] Répéter
            k = donnée ("Entrer un entier naturel non nul :")
        Jusqu'à (k>0)
Fin saisie_partie
```

Analyse de la procédure Trier :

```
Def Proc Trier (Var T : tab1 ; Var M :  tab2 ; n : octet)
Résultat = (T, M)
(T,M ) = [    ]  Pour  i de 1 à (n-1) faire
                    ind← FN Indmax(M,n,i)
                Aux1←  T[ind]
                T[ind]←  T[i]
                T[i] ←Aux1
                    Aux2 ←  M[ind]
                M[ind]←  M[i]
                M[i] ←Aux2
            Finpour
Fin Trier
```

T.D.O.L

Objet	Type	Rôle
Indmax	Fonction	Rechercher l'indice du maximum dans la partie i .. n
Ind	Octet	du tableau M Variable intermédiaire
Aux1	Chaine	Variable intermédiaire
Aux2	Réel	Variable intermédiaire

Analyse de la fonction indmax

```
Def FN Indmax (M : tab2 ; n , i : octet) : octet
Résultat  =  indmax←ind
Ind = [ ind ←i] Pour j de (i+1) à n Faire
                    Si M[j] > M[ind] Alors   ind←j
                    FinSi
Fin indmax
```

T.D.O.L

Objet	Type	Rôle
J	Octet	Compteur
ind	Octet	Indice de la valeur maximale

Analyse de la procédure affiche

```
Def  Proc Affiche (T : tab1 ; M : tab2 ; n : octet)
Résultat = [Ecrire (T[1] , "  ")]
            Pour i de 2 à N faire
                Si  M[i] <> M[i-1] Alors Retourligne
                FinSi
                Ecrire (T[i] , "  ")
            FinPour
Fin Affiche
```

N.B. : L'instruction retourligne peut être remplacée par l'instruction Ecrire().

T.D.O.L

Objet	*Type*	*Rôle*
i	*octet*	*compteur*

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION

EXAMEN DU BACCALAUREAT ---- SESSION DE JUIN 2011

SECTIONS : MATHEMATIQUES + SCIENCES EXPERIMENTALES + SCIENCES TECHNIQUES
EPREUVE : INFORMATIQUE DUREE : 1h30 COEFFICIENT : 0,5

N.B Les réponses à la <u>PARTIE I</u> doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie qui contiendra les réponses à la <u>PARTIE II</u>

PARTIE I (6 points)

Exercice 1 (4 points)

Soit l'algorithme de la fonction "**Traitement**" suivante :

```
0)  DEF FN Traitement (n : entier) : entier
1)  r ← 0
    Répéter
       r ← r + n MOD 10
       n ← n DIV 10
    Jusqu'à (n = 0)
2)  Traitement ← r
3)  Fin Traitement
```

Questions

1) Quelle est la valeur retournée par la fonction "**Traitement**" pour **n = 125 ?**

……

2) Quelle est la valeur retournée par la fonction "**Traitement**" pour **n = 458 ?**

……

3) Donner le rôle de cette fonction.

……

……

Exercice 2 (2 points)

Répondre par **Vrai** si la proposition est correcte ou par **Faux** dans le cas contraire.

Proposition	Réponse
Un tableau de réels peut être rempli par des entiers.	
Le compteur d'une structure répétitive complète doit être de type scalaire.	
Les opérateurs **DIV** et **MOD** peuvent être appliqués sur les nombres réels.	
Efface (ch,longueur(ch)-1,2) efface les deux derniers caractères de la chaîne **ch**.	

PARTIE II (14 points)

*Un entier **n** de **4** chiffres est dit **valable**, si ses trois derniers chiffres sont des multiples de son chiffre des milliers.*

Exemple : L'entier **2648** est **valable** car son chiffre des milliers est **2** et il est suivi par les chiffres **6**, **4** et **8** qui sont tous multiples de **2**.

On se propose d'écrire un programme qui permet de lire un entier positif **n** composé de **4** chiffres puis d'afficher s'il est **valable** ou non.

Exemple 1 : Si **n= 2888** alors le programme affichera : ***Cet entier est valable***.

Exemple 2 : Si **n= 2179** alors le programme affichera : ***Cet entier n'est pas valable***.

Questions

1) Analyser le problème en le décomposant en modules et en déduire l'algorithme du programme principal.
2) Analyser chacun des modules envisagés et en déduire les algorithmes correspondants.

CORRIGÉ

PARTIE I (6 points)

Exercice 1 (4 points)

1) 8

2) 17

3) La fonction retourne la somme des chiffres qui composent l'entier **n**.

Exercice 2 (2 points)

Proposition	Réponse
Un tableau de réels peut être rempli par des entiers.	***Vrai***
Le compteur d'une structure répétitive complète doit être de type scalaire.	***Vrai***
Les opérateurs **DIV** et **MOD** peuvent être appliqués sur les nombres réels.	***Faux***
Efface (ch,longueur(ch)-1,2) efface les deux derniers caractères de la chaîne **ch**.	***Vrai***

PARTIE II (14 points)

Analyses : 8 points, Algorithme 4 points, paramètres et objets Locaux et Globaux : 2 points

ANALYSE DU PROGRAMME PRINCIPAL :

NOM = Valable
2. Résultat= PROC Affiche(n)
1. PROC Saisie(n)
3. Fin Valable

T.D.O

Objet	Type/Nature	Rôle
n	Entier	Le nombre d'éléments
Affiche	Procédure	Procédure de d'affichage
saisie	Procédure	Procédure de saisie

ALGORITHME DU PROGRAMME PRINCIPAL :

0) Début Valable
1) PROC Saisie(n)
2) PROC Affiche(n)
3) Fin Valable

ANALYSE DE LA PROCEDURE SAISIE :

DEF PROC Saisie (var m : entier)
2. Résultat= (m)
1. m= [] Répéter
m = donnée
Jusqu'à (m >= 1000) et (m<=9999)
3. Fin Saisie.

ALGORITHME DE LA PROCEDURE SAISIE :

0) DEF PROC Saisie (var m : entier)
1) Répéter
Lire(m)
Jusqu'à (m >= 1000) et (m<=9999)
2) Fin Saisie

ANALYSE DE LA PROCEDURE AFFICHE :

DEF PROC Affiche (p : entier)
2) Résultat = Affich
1) Affich = [] Si Valable (p) = Vrai Alors
Ecrire (p, "est valable")
Sinon
Ecrire (p, "n'est pas valable")
FinSi
3) Fin Affiche

ALGORITHME DE LA PROCEDURE AFFICHE

0) DEF PROC Affiche (p : entier)
1) Si Valable(p) = vrai Alors
Ecrire (p, "est valable")
Sinon
Ecrire (p, "n'est pas valable")
FinSi
2) Fin Affiche

ANALYSE DE LA FONCTION VALABLE :

DEF FN Valable (k : entier): booléen
3) Résultat = Valable
2) Valable = [] Si (c mod m = 0) et (d mod m = 0) et (u mod m = 0) Alors
Valable ← vrai
Sinon
Valable ← faux
FinSi

1) m← k div 1000
c← k mod 1000 div 100
d← k mod 100 div 10
u← k mod 10

4) Fin Valable.

T.D.O

Objet	Type/Nature	Rôle
m	entier	Chiffre des milliers.
c	entier	Chiffre des centaines.
d	entier	Chiffre des dizaines.
u	entier	Chiffre des unités.

ALGORITHME LA FONCTION VALABLE :

0) DEF FN Valable (k: entier) : booléen
1) m← k div 1000
c← k mod 1000 div 100
d← k mod 100 div 10
u← k mod 10
2) Si (c mod m = 0) et (d mod m = 0) et (u mod m = 0) Alors
Valable ← vrai
Sinon
Valable ← faux
FinSi
3) Fin Valable

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION

EXAMEN DU BACCALAUREAT - SESSION DE JUIN 2010

SECTIONS : Mathématiques + Sciences Expérimentales + Sciences Techniques

EPREUVE : INFORMATIQUE **DUREE** : 1,5 h **COEFFICIENT** : 0,5

Important :

- **Le sujet comporte 3 pages numérotés 1/3, 2/3 et 3/3.**
- **La réponse à la "Partie I" se fera sur les mêmes pages 1/3 et 2/3 de ce sujet qui doivent être remises à la fin de l'épreuve.**
- **La réponse à la "Partie II" est à développer sur une feuille de copie.**

Partie I (6 points)

Exercice 1 (3 points)

On suppose qu'un programme principal contient trois sous programmes (une procédure **Proc1**, une fonction **Fonct** et une procédure **Proc2**).

Compléter le tableau suivant par un exemple d'appel de chacun des sous programmes au niveau du programme principal, en se basant sur les entêtes et sur la liste des variables globales disponibles.

Entête du sous programme	**Variables globales**	**Exemple d'appel du sous programme dans le programme principal**
Procédure Proc1 (VAR m,n:entier; z:réel)	a,b : entier x : réel car : caractère Mot : chaîne	
Fonction Fonct(n:entier; Ch:chaîne):caractère		
Procédure Proc2 (Ch:chaîne ; VAR c:caractère)		

Exercice 2 (3 points)

Soit le programme Pascal suivant :

```
Program ESSAI ;
Uses wincrt ;
Var  y : integer ;

Function Fonct (a : integer): Char;
Begin
  Fonct := Chr(2*a);
End;

Procedure Proc  ;
Var
  m : Char;
Begin
  m:=Fonct(y);
  Writeln(m);
End;

Begin
  Readln(y);
  Proc;
End.
```

Questions

1) Compléter le tableau suivant par la nature de chaque objet utilisé (objet local ou objet global)

Objet	Nature
y	
m	

2) Pour les objets **y**, **m**, **Proc** et **Fonct**, compléter le tableau ci-dessous en mettant une croix (**x**) dans la case correspondante si l'objet est visible par le programme principal "**ESSAI**" ou par les sous programmes :

	Programme principal	Sous programmes	
Objet	**ESSAI**	**Proc**	**Fonct**
y			
m			
Proc			
Fonct			

Partie II (14 points)

On se propose d'écrire une analyse et un algorithme d'un programme "**Tri**" qui permet de remplir un tableau **T** par **n** entiers distincts puis de former et d'afficher un autre tableau **Res** qui va contenir les **n** entiers du tableau **T** classés en ordre croissant selon le principe suivant :

Pour chaque élément du tableau **T**

1) Déterminer le nombre **Nbr** d'éléments de **T** qui lui sont inférieurs ou égaux.
2) Placer cet élément dans la position **Nbr** du tableau **Res**.

Exemple : pour les éléments du tableau suivant :

T	3	14	0	9	17	5	8	4
	1	2	3	4	5	6	7	8

- L'entier **T[1] = 3** a **2** éléments qui lui sont inférieurs ou égaux (**3 et 0**), il sera placé dans la position **2** du tableau **Res**.

Res		3						
	1	2	3	4	5	6	7	8

- L'entier **T[2] = 14** a **7** éléments qui lui sont inférieurs ou égaux (**3 , 14 , 0 , 9 , 5 , 8** et **4**), il sera placé dans la position **7** du tableau **Res**.

Res		3					14	
	1	2	3	4	5	6	7	8

- ainsi de suite pour les autres éléments …

Le tableau **Res** aura les éléments placés dans un ordre croissant comme suit :

Res	0	3	4	5	8	9	14	17
	1	2	3	4	5	6	7	8

Questions :

1. Analyser le problème en le décomposant en modules et déduire un algorithme du programme principal.
2. Analyser chacun des modules proposés.

EXAMEN DU BACCALAUREAT - SESSION DE JUIN 2010

SECTIONS : Mathématiques + Sciences Expérimentales + Sciences Techniques

Corrigé du sujet théorique d'informatique

Partie I (6 points)

Exercice 1 (3 points)

On suppose qu'un programme principal contient trois sous programmes (une procédure **Proc1**, une fonction **Fonct** et une procédure **Proc2**).
Compléter le tableau suivant par un exemple d'appel de chacun des sous programmes au niveau du programme principal, en se basant sur les entêtes et sur la liste des variables globales disponibles.

Entête du sous programme	Variables globales	Exemple d'appel du sous programme dans le programme principal
Procedure Proc1 (VAR X,Y:integer; Z:real);	A,B : integer M : real L : char Mot : string	***Proc1(A , B , M)*** ***ou*** ***Proc1(B , A , M)***
Function Fonct(X:integer; Z:string):Char;		***L := Fonct(A , Mot)*** ***ou*** ***L := Fonct(B , Mot)***
Procedure Proc2 (Ch:string ; VAR C:char);		***Proc2(Mot , L)***

Exercice 2 (3 points)

Soit le programme Pascal suivant :

```
Program ESSAI ;
Uses wincrt ;
Var  y : integer ;

Function Fonct (a : integer): Char;
Begin
Fonct := Chr(2*a);
End;

Procedure Proc  ;
Var
m : Char;
Begin
m:=Fonct(y);
Writeln(m);
End;

Begin
Readln(y);
Proc;
End.
```

Questions

1) Compléter le tableau suivant par la nature de chaque objet utilisé (objet local ou objet global)

Objet	Nature
y	***Global***
m	***Local***

2) Pour les objets **y**, **m**, **Proc** et **Fonct**, compléter le tableau ci-dessous en mettant une croix (**x**) dans la case correspondante si l'objet est visible par le programme principal "**ESSAI**" ou par les sous programmes :

	Programme principal	Sous programmes	
Objet	**ESSAI**	**Proc**	**Objet**
y	***x***	***x***	***x***
m		***x***	
Proc	***x***		
Fonct	***x***	***x***	

Partie II (14 points)

On se propose d'écrire une analyse et un algorithme d'un programme "**Tri**" qui permet de remplir un tableau **T** par **n** entiers distincts puis de former et d'afficher un autre tableau **Res** qui va contenir les **n** entiers du tableau **T** classés en ordre croissant selon le principe suivant :

Pour chaque élément du tableau **T**

1) Déterminer le nombre **Nbr** d'éléments de **T** qui lui sont inférieurs ou égaux.
2) Placer cet élément dans la position **Nbr** du tableau **Res**.

Exemple : pour les éléments du tableau suivant :

T	3	14	0	9	17	5	8	4
	1	2	3	4	5	6	7	8

- L'entier **T[1] = 3** a **2** éléments qui lui sont inférieurs ou égaux (**3 et 0**), il sera placé dans la position **2** du tableau **Res**.

Res		3						
	1	2	3	4	5	6	7	8

- L'entier **T[2] = 14** a **7** éléments qui lui sont inférieurs ou égaux (**3 , 14 , 0 , 9 , 5 , 8** et **4**), il sera placé dans la position **7** du tableau **Res**.

Res		3					14	
	1	2	3	4	5	6	7	8

- ainsi de suite pour les autres éléments …

Le tableau **Res** aura les éléments placés dans un ordre croissant comme suit :

Res	0	3	4	5	8	9	14	17
	1	2	3	4	5	6	7	8

Questions :

1. Analyser le problème en le décomposant en modules et déduire un algorithme du programme principal.
2. Analyser chacun des modules proposés.

Analyse du Programme Principal

Résultat = Affiche Tableau trié
[]Pour i de 1 à N faire
    Ecrire("RES[",i, "]= ",RES[i])
Traitement :
  2) PROC Trier(N,T,RES)
  1) PROC lecture(N,T)

Types

Types
TAB = Tableau de 50 entiers

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
N	Entier	Dimension du tableau.
T	Tab	Tableau d'entiers.
RES	Tab	Tableau trié
i	entier	compteur
Trier	procédure	Permet de trier le tableau T dans RES
Lecture	Procédure	Permet la saisie contrôlée de N et le remplissage du tableau T avec contrôle.

Analyse de la procédure Lecture :

```
DEF PROC Lecture(var N:entier;Var T:tab)
2) T= []répéter
        distinct← Vrai
        [] Pour  i de  1 à N  Faire
            T= Donnée ("T[",i, "]= ")
          FinPour
        [] Pour  i de  1 à N  Faire
             [] Pour  j de i+1 à n Faire
                  []Si t[i]= t[j] alors  distinct← faux  FINSI
             FinPour
         FinPour
        [] Si  NON distinct alors Ecrire("Saisir des éléments distincts FINSI
   Jusqu' à   distinct= Vrai
1) N=  []Répéter
        N= donnée("N= ")
        Jusqu'à  (N>=2) et (N<=100)
3) Fin Lecture
```

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
Distinct	Booléen	Pour s'assurer que le tableau contient des éléments distincts
i	Entier	Compteur
j	Entier	Compteur

Analyse de la procédure Trier:

```
0) DEF PROC Trier(N:entier;T:tab; var RES:TAB)
0) []Pour  i de 1 à  n faire
        [ K←0] Pour j de 1 à n do
              []SI t[j] < t[i] Alors  k← k+1    Finsi
                FinPour
              RES[K+1]← t[i]
   FinPour
1) Fin Trier
```

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
i	Entier	Compteur
j	Entier	Compteur
K	Entier	L'iIndice de l'emplacement des bonnes places des éléments dans le tableau RES

LES ALGORITHMES :

Algorithme du programme Principal

```
0) Début Prog_Princ
1) PROC Lecture(N,T)
2) PROC Trier(N,T,RES)
3) Pour i de 1 à N Faire
        Ecrire("RES[",i, "]= ",RES[i])
3) Fin Prog_Princ
```

Algorithme de la procédure Lecture

```
0) DEF PROC Lecture(var N:entier;Var T:tab)
1) Répéter
        Ecrire("N= ")
        Lire(N)
   Jusqu'à (N>=2) et (N<=100)
2) Répéter
        Distinct← Vrai
        Pour i de 1 à N Faire
            Ecrire("T[",i, "]= ")
            Lire(T[i])
        FinPour
        Pour i de 1 à N Faire
            Pour j de i+1 à n Faire
                SI t[i]= t[j] alors Distinct← Faux FINSI
            FinPour
        FinPour
        SI (NON distinct) Alors Ecrire("Saisir des éléments distincts") Finsi
  Jusqu' à (distinct= Vrai)
3) Fin Lecture
```

Algorithme de la procédure Trier

```
1) DEF PROC Trier(N:entier;T:tab; var RES:TAB)
2) Pour i de 1 à n faire
        K←0
        Pour j de 1 à n do
            Si T[j] < T[i] Alors
                k← k+1
            FinSi
        FinPour
        RES[K+1]← T[i]
   FinPour
3) FIN Trier
```

Le programme Pascal

```
Program Welcome;
Uses WinCrt;
Type
    TAB = Array[1..100] of integer;
Var
    N,j,k,i : integer;
    T,RES : TAB;
    distinct: boolean;
```

```
Procedure Lecture(Var N:integer ; var T:TAB);
Begin
        Repeat
                 Write('N= ');
                Readln(N);
        Until (N>=2) And (N<=100);
        Repeat
                distinct:= true;
                For i:= 1 to N do
                   Begin
                        Write('T[',i,']= ');
                        Readln(T[i]);
                   End;
                For i:=1 to n do
                        For j:=i+1 to n do
                                if t[i]= t[j] Then  distinct := false ;
                if (Not distinct) Then Writeln(' Saisir des éléments distincts');
        Until distinct= True;
End;

Procedure Trier(N:integer;T:tab;VAR RES:TAB) ;
Begin
    For i:=1 to N do
        Begin
                 k:=0;
                For j:= 1 to n do
                        If t[j] < t[i] Then k:= k+1 ;
                RES[K+1]:= t[i];
        End;
End;

Begin
        Lecture(N,T);
        Trier(N,T,RES);
        For i:= 1 to N do
                Writeln('RES[',i,']= ',RES[i]);
End.
```

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ET DE LA FORMATION	EXAMEN DU BACCALAURÉAT SESSION DE JUIN 2009	
SECTIONS :	MATHÉMATIQUES + SCIENCES EXPÉRIMENTALES + SCIENCES TECHNIQUES	
EPREUVE : INFORMATIQUE	DURÉE : 1h30	COEFFICIENT : 0,5

Les réponses à la partie I doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie.

PARTIE I (8 points)

Exercice N°1 : (1.5 points)
Soit la fonction Pascal suivante :

```
Function Somme (a,b :integer) : integer ;
Var     p: integer;
Function Produit (n : integer): integer;
        Var     q,r : integer;
        Begin
          q := n div 3 ;
          r  := n mod 3 ;
        Produit := q * r;
        End;
Begin
   p := Produit(a) + Produit(b) ;
  Somme :=p ;
End ;
```

Indiquer pour chaque objet s'il est reconnu par la fonction **Somme,** la fonction **Produit** ou les deux fonctions en même temps en mettant dans la case correspondante la lettre **O** (Oui) si l'objet est reconnu ou la lettre **N** (Non) s'il n'est pas reconnu.

Objet	Reconnu par la fonction	
	Somme	Produit
p		
q		
r		

Exercice N°2 : (3.5 points)

Soient les déclarations Pascal suivantes :

```
Type
    jours_semaine = (Lundi, Mardi, Mercredi, Jeudi, Vendredi, Samedi, Dimanche) ;
    langues_etrangeres = (Italien, Anglais, Espagnol, Allemand) ;
Var
    aujourdhui, jour : jours_semaine ;
    langue : langues_etrangeres ;
    res : boolean ;
    n : integer ;
```

Compléter le tableau ci-dessous en mettant une croix dans la **colonne valide** ou dans la **colonne non valide** pour chaque instruction. Justifier votre réponse pour les instructions non valides.

Instruction	Valide	Non valide	Justification
Readln(jour) ;			
langue :=Anglais ;			
aujourdhui := Dimanche ;			
Writeln(langue) ;			
res := aujourdhui<jour ;			
n := ord(langue) ;			

Exercice N°3 : (3 points)

Soit la fonction Pascal suivante :

```
Function Essai (ch: string; nb, p: integer): string;
Var
  i : ..... ;      r : ..... ;
Begin
      r := '';
      For i:=1 to length (ch) do
            if i in [p..p+nb-1] then r := r + ch[i];
      Essai := r;
End;
```

Questions:

1. Compléter les pointillés par des types appropriés.
2. Donner le résultat de cette fonction pour les paramètres effectifs suivants :

Ch = 'Protocole' **p**= 3 et **nb**= 4

.....

3. En utilisant les paramètres effectifs de la fonction **Essai**, donner la fonction prédéfinie Pascal qui fournit le même résultat.

.....

.....

PARTIE II (12 points)

On se propose d'écrire un programme permettant de remplir deux tableaux **V1** et **V2** respectivement par **N** et **M** entiers (avec N et M deux entiers de l'intervalle **[2..20]** et les éléments de **V1** et **V2** sont saisis dans un ordre strictement croissant), puis de fusionner les éléments de ces deux tableaux dans un tableau **V3** en éliminant les redondances et en gardant l'ordre croissant des éléments. Enfin, le programme affichera les éléments du tableau **V3**.

Exemple

SI **N** = 5, **M** = 7 et les éléments des deux tableaux **V1** et **V2** sont :

V1	**4**	7	8	9	**12**

V2	0	2	**4**	**7**	10	**12**	28

Alors le tableau **V3** contiendra les éléments suivants :

V3	0	2	4	7	8	9	10	12	28

Travail demandé :

1. Analyser le problème en le décomposant en modules.
2. Analyser chacun des modules proposés.
3. En déduire un algorithme du programme principal ainsi que ceux des modules envisagés.

Correction Informatique Section Sections Scientifiques
Session Juin 2009

PARTIE I (8 points)

Exercice N°1 : (1,5 points = 0,25 x 6)

Règles à appliquer :

Si la définition d'un module **M** nécessite la déclaration de **X** objets notés $\mathbf{O_i}$ et de **Y** sous-modules notés $\mathbf{M_i}$, alors les **X** objets $\mathbf{O_i}$ seront reconnus par le module **M** mais également par les **Y** sous-modules $\mathbf{M_i}$.

Application de la règle sur l'objet p :

La définition du module **Somme** nécessite la déclaration de l'objet **p** et d'une fonction **Produit**, donc **p** est reconnu par le module **Somme** mais également par le sous-module **Produit**.

Application de la règle sur les objets q et r :

La définition du module **Produit** nécessite la déclaration des objets **q** et **r** et d'aucun sous-module, donc **q** et **r** sont reconnus par le module **Produit** et **ne seront pas reconnus ailleurs**.

Objet	Reconnu par la fonction	
	Somme	Produit
p	O	O
q	N	O
r	N	O

NB: On n'acceptera que les réponses O/N et Oui/Non

Exercice N°2 : (3,5 points = 0,5 par croix et 0,25 par justification = 0.5*6 + 0.25*2)

Readln(jour) ; Cette instruction **n'est pas valide** car, on ne peut pas lire une variable de type scalaire énuméré.

langue :=Anglais ; Cette instruction est **valide**. En effet, **langue** étant une variable de type **langues_etrangeres** et **Anglais** étant une des valeurs de cet ensemble, l'affectation est possible.

aujourdhui := Dimanche ; Cette instruction est **valide**. En effet, **aujourdhui** étant une variable de type **jour_semaine** et **dimanche** étant une des valeurs de cet ensemble, l'affectation est possible.

Writeln(langue) ; Cette instruction **n'est pas valide** car, on ne peut pas afficher une variable de type scalaire énuméré.

res := aujourdhui<jour ; Cette instruction est **valide**. En effet, il s'agit d'affecter à une variable de type logique (**res**) le résultat de la comparaison **aujourdhui<jour,** qui est un résultat de type **logique**.

Exemple : *si **aujourdhui** vaut **Dimanche** et **jour** vaut **Lundi**, alors aujourdhui<jour est faux, car ord(Dimanche) > ord(jour)*

n := ord(langue) ; Cette instruction est **valide**. En effet, il s'agit d'affecter à une variable de type entier (**n**) le numéro d'ordre (**ord**) du contenu de la variable **langue**. Ce numéro est un entier.

Exemple : *si **langue** vaut **Anglais**, alors ord(langue) = 1 et n vaut 1.*

Exercice N°3 (3 points = 1 + 1 + 1)

1. *Function Essai (ch: string; nb, p: integer): string;* ***(1 pt=2*0,5)***

```
Var
  i : Byte ;      r : string ;   Il est possible d'accepter le type Integer pour la variable i.
Begin
      r := '';
      For i:=1 to length (ch) do
            if i in [p..p+nb-1] then r := r + ch[i];
      Essai := r;
End;
```

2. Le résultat de cette fonction pour les paramètres effectifs suivants :

Ch = 'Protocole' **p**= 3 et **nb**= 4

Ch	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'
Nb	4	4	4	4	4				
P	3	3	3	3	3				
I	1	2	3	4	5	6	7	8	9
r	''	''	'o'	'ot'	'oto'	'otoc'	'otoc'	'otoc'	'otoc'

La fonction retourne la valeur : **otoc (*1 pt)***

3. la fonction **essai** renvoie une sous-chaîne de **nb** caractères et formée à partir d'une chaîne notée **Ch** en commençant à la position **p**. Il s'agit de la fonction prédéfinie **COPY (Ch, p, nb)**. (***1 pt)***

PARTIE II (12 points)

Analyse (8 Pts)	**Algorithme**
• On acceptera toute forme d'analyse descendante. • On acceptera toute solution correcte.	• *Pour être évalué, l'algorithme doit présenter un contenu en relation avec le problème demandé*
PP (1,5 pts) • Cohérence=0,5 • Modularité= 1 Saisie (1,5 pts) • Taille = 0,25*2	• PP (1 pt) Saisie (1 pt)

• Tableau= 0,25*4 Fusion (3 pts) • Union = 1 • Ordre = 1 • Redondance = 1 Affichage (1 pts=2*0,5) Divers (1 pts) • Entête des modules (mode de passage, ordre et types de paramètres) • T.D.O et des nouveaux types	Fusion (1,5 pt) Affichage (0,5 pt)

Nom = Distinct		
S	**L.D.E.**	**O.U.**
4	Résultat = Proc Afficher (K, V3)	K, V3
3	(K, V3) = Proc Fusion (N, M, V1, V2, K, V3)	Afficher
2	(N, V1) = Proc Saisie (N, V1)	Fusion
1	(M, V2) = Proc Saisie (M, V2)	N, V1
5	Fin	M, V2, saisie

Tableau de déclaration des nouveaux types

Type
Tabd =Tableau de 20 entiers
Tabr = Tableau de 40 entiers

T.D.O.G

Nom	**Type**	**Rôle**
K	Entier	Longueur du tableau final V3
N	Entier	Longueur du tableau V1
M	Entier	Longueur du tableau V2
V1	Tabd	Tableau donné
V2	Tabd	Tableau donné
V3	Tabr	Tableau final
Afficher	Procédure	Affiche le tableau final
Saisie	Procédure	Saisie d'un tableau et sa longueur
Fusion	Procédure	Fusionne V1 et V2 dans V3 sans redondance

DEF PROC Saisie (var Nb : entier ; var T :Tabd)		
S	**L.D.E.**	**O.U.**
1	Résultat = Nb, T Nb = [ ] Répéter	

Commentaires

• Validation de la

	Nb = donnée Jusqu'à Nb dans [2..20] T= [T[1] = donnée]	
2	Pour i de 2 à Nb faire Répéter T[i]=donnée Jusqu'à T[i] > T[i-1] Fin pour	i
3	Fin	

taille des tableaux

- Lecture de l'élément n°1
- Tous les autres éléments doivent être dans un ordre strictement croissant

T.D.O.L

Nom	Type	Rôle
i	**Entier**	**Compteur**

DEF PROC Fusion (N, M : Entier ; V1, V2 : Tabd ; Var K : Entier ; Var V3 : Tabr)

S	L.D.E.	O.U.
1	Résultat = K, V3 (K,V3)=[i← 1, j← 1, K← 0] Répéter K← K+1 Si V1[i]<V2[j] alors V3[K]← V1[i] i ← i+1 Sinon Si V1[i]=V2[j] alors V3[K]← V1[i] i← i+1 j← j+1 Sinon V3[K]← V2[j] j ← j+1 FinSi FinSi Jusqu'à (i>N) ou (j>M)	i, j
2	Si j>M Alors Pour c de i à N faire K← K+1 V3[K] ← V1[c] Fin pour Sinon Pour c de j à M faire K← K+1 V3[K] ← V2[c] Fin Pour	c

Commentaires

- L'élément de **V1** est plus petit que celui de **V2**, on range **V1[i]** et on avance dans **V1**
- Ecriture d'un seul élément (**sans redondance**)
- L'élément de **V2** est plus petit que celui de **V1**, on range **V2[j]** et on avance dans **V2**
- Répétition du traitement jusqu'à **la fin de V1 ou de V2**.
- Ecriture des éléments de **V1** (**car c'est la fin de v2**)
- Ecriture des éléments de **V2** (**car c'est la fin de V1**)

3	FinSi Fin	

T.D.O.L

Nom	Type	Rôle
i	Entier	compteur
j	Entier	compteur
c	Entier	compteur

DEF PROC afficher (Nb : entier ; T :tabr)		
S	**L.D.E.**	**O.U.**
1 **2**	Résultat = [ ] Pour i de 1 à Nb faire Ecrire (T[i]) Fin pour Fin	i

T.D.O.L

Nom	Type	Rôle
I	Entier	compteur

Les algorithmes

Programme Principal

0) Début Distinct
1) Proc Saisie (N, V1)
2) Proc Saisie (M, V2)
3) Proc Fusion (N, M, V1, V2, K, V3)
4) Proc Afficher (K, V3)
5) Fin Distinct

Procédure Saisie

0) **DEF PROC saisie (Var Nb : Entier ; Var T :Tabd)**
1) Répéter
Lire (Nb)
Jusqu'à Nb dans [2..20]
lire(T[1])
2) Pour i de 2 à Nb faire
Répéter
Lire (T[i])
Jusqu'à T[i] >T[i-1]
Fin pour
3) Fin saisie

Procédure Fusion

0) **DEF PROC Fusion (N, M : entier ; V1,V2 : tabd ; var K : entier ; var V3 : tabr)**
1) i← 1, j← 1, K← 0
Répéter
K← K+1

```
Si V1[i]<V2[j] alors
                    V3[K]← V1[i]
                    i ← i+1
Sinon
  Si V1[i]=V2[j] alors
                    V3[K]← V1[i]
                    i ← i+1
                    j ← j+1
  Sinon
   V3[K]← V2[j]
   j ← j+1
  FinSI
FinSi
Jusqu'à (i>N) ou (j>M)
Si j>M Alors
                Pour c de i à N faire
                K← K+1
                 V3[K] ← V1[c]
                Fin pour
Sinon
Pour c de j à M faire
K← K+1
V3[K] ← V2[c]
Fin Pour
FinSi
Fin Fusion
```

2)

3)

Procédure Afficher

```
0)  DEF PROC Afficher ( Nb : entier ;  T :tabr)
1)  Pour i de 1 à Nb faire
    Ecrire (T[i])
    Fin pour
2)  Fin Afficher
```

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ET DE LA FORMATION EXAMEN DU BACCALAUREAT SESSION DE JUIN 2008			NOUVEAU REGIME
SECTIONS :	MATHEMATIQUES + SCIENCES EXPERIMENTALES + SCIENCES TECHNIQUES		
EPREUVE :	INFORMATIQUE	DUREE : 1 h 30	COEFFICIENT : 0,5

Les réponses à la partie 1 doivent être rédigées sur cette même feuille qui doit être remise à la fin de l'épreuve avec la feuille de copie.

PARTIE I (8 points)

Exercice 1 : (3 points)

Compléter le tableau ci-dessous, par les déclarations Pascal adéquates :

Description	Déclaration en Pascal (préciser le mot clé adéquat : CONST, TYPE, VAR, etc.)
Une chaîne **ch** de 20 caractères au maximum.	
Un type **saison** contenant les identificateurs suivants : automne, hiver, printemps, ete.	
Un tableau **V** dont les indices sont de type caractère pouvant contenir 20 chaînes.	
Une constante **message** de valeur « Bonne chance ».	

Exercice 2 : (2 points)

Soit la fonction Existe dont l'algorithme est donné ci-après :

```
0) DEF FN Existe( n:_____; T :_____; x :_____) : _____
1) Trouve ← _____.
2) i ←0
3) Répéter
        _____.
        Si (T[i] = x) alors
            Trouve ← vrai
        finsi
4) Jusqu'à ( _____ ) ou (_____)
5) _____.
6) Fin Existe
```

Compléter l'algorithme de la fonction **Existe,** dont les paramètres sont **x**, **n** et **T**, qui permet de vérifier l'existence d'un élément **x** dans un tableau **T de n réels**.

Exercice 3 : (3 points)

Compléter les affectations suivantes par une valeur d'opérande ou d'opérateur permettant d'obtenir dans chacun des cas, la valeur voulue de Y:

Affectation	Valeur de Y
`Y := round(99,51) = ..... ;`	True
`Y := (upcase('a') in ['A' .. 'Z']) and (..... in [1..10]);`	True
`Y := length('PASCAL') mod 4 ..... 2 ;`	True
`Y := random(4) ..... 4;`	True
`Y := pred('D') = chr(ord('.....') + 1);`	True
`Y := copy('informatique',1,4) ..... 'info';`	False

PARTIE II: (12 points)

On se propose d'écrire un programme qui permet de saisir une chaîne non vide de 100 caractères au maximum et qui détermine et affiche les informations suivantes :

- Le nombre total de caractères dans la chaîne,
- le nombre d'occurrences de chaque lettre alphabétique figurant dans la chaîne (sans distinction entre minuscule et majuscule),
- le nombre total de caractères non alphabétiques.

N.B. : On suppose que les lettres accentuées ne sont pas considérées comme des lettres alphabétiques.

Exemple :
Pour la chaîne: "Ceci est une épreuve du baccalauréat."
Le programme devra afficher le résultat suivant :
Votre texte comporte 37 caractères dont :
4 fois la lettre A
1 fois la lettre B
4 fois la lettre C
1 fois la lettre D
5 fois la lettre E
1 fois la lettre I
1 fois la lettre L
1 fois la lettre N
1 fois la lettre P
2 fois la lettre R
1 fois la lettre S
2 fois la lettre T
4 fois la lettre U
1 fois la lettre V
et 8 caractères non alphabétiques.

Questions :

1. Quelles sont les structures de données à utiliser pour résoudre ce problème ? Justifier le choix de chaque structure proposée.
2. Analyser le problème en le décomposant en modules.
3. Analyser chacun des modules envisagés dans l'analyse du programme principal.
4. Déduire de ce qui précède l'algorithme du programme principal ainsi que les algorithmes des modules envisagés.

REPUBLIQUE TUNISIENNE MINISTERE DE L'EDUCATION ET DE LA FORMATION *** EXAMEN DU BACCALAUREAT SESSION 2008 (SP)	Sections : Math. + Tech. + Sc.Exp. EPREUVE THEORIQUE D'INFORMATIQUE **SOLUTION** DUREE : 1 h COEFFICIENT : 0.5

PARTIE I (8 points)

Exercice 1 : (3 points = 4 x 0,75)

Compléter le tableau ci-dessous, par les déclarations Pascal adéquates :

- 0.25 / mot clé + 0.5 pour la suite de la déclaration

Description	Déclaration en Pascal (Préciser le mot clé adéquat : CONST, TYPE, VAR, etc.)	Remarques
Une chaîne **ch** de 20 caractères au maximum.	**Var** **ch : string[20] ;**	
Un type **saison** contenant les identificateurs suivants : Automne, Hiver, Printemps, ete.	**Type saison = (Automne, Hiver, Printemps, ete) ;**	**L'ordre est important**
Un tableau **V** dont les indices sont de type caractère pouvant contenir 20 chaînes.	**Var** **V : array['A'. . 'T'] of string ;**	**On acceptera toute combinaison équivalente de 20 éléments**
Une constante **message** de valeur « Bonne chance ».	**Const** **Message='Bonne chance' ;**	

Exercice 2 : (2 points = 8 x 0,25)

Soit la fonction Existe dont l'algorithme est donné ci-dessous :

0) DEF FN Existe (n: entier ; T : **tab ;** x : **réel**) : **booléen**
1) Trouve ← **faux**
2) i ←0
3) Répéter
   i ← i +1
   Si (T[i] = x) alors
      Trouve ← vrai
   FinSi
4) Jusqu'à (**Trouve**) ou (**i =n**)
5) Existe ← Trouve
6) Fin Existe

- On acceptera que le type de la fonction soit **chaîne** avec la séquence **5)** une structure conditionnelle cohérente.

Compléter l'algorithme de la fonction **Existe,** dont les paramètres sont x, n et T, qui permet de vérifier l'existence d'un élément x dans un tableau **T** de n réels.

Exercice 3 : (3 points)

Compléter les affectations suivantes par une valeur d'opérande ou d'opérateur permettant d'obtenir dans chacun des cas, la valeur de Y voulue :

Affectations	**Valeur de Y**	
Y := round(99,51) = **100** ;	True	
Y := (upcase('a') in ['A' .. 'Z']) and (**1** in [1..10]);	True	Toute valeur comprise entre 1 et 10
Y := length('PASCAL') mod 4 = 2 ;	True	
Y := random(4) < 4;	True	Y := random(4) <> 4;
Y := pred('D') = chr(ord('**B**') + 1);	True	
Y := copy('informatique',1,4) <> 'info';	False	

NB. : Certaines affectations ont plusieurs solutions correctes.

Solution de la partie 2 : (12 points)

1) Structures données (1.5 pt)

On évaluera les structures de données de base et non les compteurs (la chaîne à traiter, le conteneur des fréquences des lettres, la variable pour les caractères non alphabétiques)

La chaîne à traiter sera nommée ch.

Les résultats du problème peuvent être récupérées dans un tableau LET pour les lettres et dans une variable NLET pour les autres.

Le tableau LET est de type TLET qui est un tableau de 26 entiers et qui aura comme indices la plage des lettres alphabétiques majuscules "A" à "Z".

2) Analyse du programme principal et des modules

Points clés de l'analyse	Barème
Programme principal (modularité + cohérence)	**1.5**
Saisie contrôlée de ch	**1**
Le comptage	**2 + 0.5 (initialisation)**
Premier affichage	**0.25**
Deuxième affichage	**1**
Troisième affichage	**0.25**
TDO et TNT	**0.5**

	NOM = nbre_lettres	
S	**L.D.E.**	**O.U.**
3	Résultat = (Ecrire(" Votre texte comporte " , longueur(ch), " caractères dont : "),	afficher
2	PROC afficher(LET), écrire(" et " , NLET, " caractères non alphabétiques.")	decompte
1	(LET, NLET) = PROC decompte(ch, LET, NLET)	LET
	ch = REPETER	NLE
	DONNEE ("Chaîne à traiter :")	Ch
	JUSQU'A (longueur (ch) dans [1..100])	
4	Fin nbre_lettres	

Tableau de déclaration des nouveaux types

TYPE	
TLET	tableau de 26 entiers et dont les indices sont "A", "B", … , "Z"

Tableau de déclaration des objets

NOM	TYPE	ROLE
afficher	procédure	- permet d'afficher les éléments du tableau avec les commentaires relatifs
decompte	procédure	- permet de déterminer la décomposition de la chaîne
LET	TLET	- ses éléments comportent respectivement le nombre d'une lettre de la chaîne ch.
NLET	Entier	- le nombre de caractères non alphabétiques
ch	Chaine[100]	- chaîne à traiter

3) Analyse des modules

Analyse du module **afficher**

	DEFPROC afficher(TL : TLET)	
S	**L.D.E.**	**O.U.**
1	Résultat = affiche_lettres affiche_lettres = [ ] Pour c de "A" à "Z" faire [ ] Si (TL[c] ≠ 0) alors Ecrire(TL[c], " fois la lettre ", c) FinSi FinPour	c
2	Fin afficher	

Tableau de déclaration des objets locaux

NOM	TYPE	ROLE
c	caractère	compteur et en même temps indice

Analyse du module **decompte**

	DEFPROC decompte(cht : chaine[100] ; **VAR** TL : TLET ; **VAR** NLT : entier)	
S	**L.D.E.**	**O.U.**
1	Résultat = (TL, NLT) (TL, NLT) = [l←Long(cht), NLT ← 0, PROC init(TL),] Pour i de 1 à l Faire [c ← Majus(cht[i])] Si (c dans ["A".."Z"]) alors TL[c] ← TL[c] + 1 Sinon NLT ← NLT + 1 FinSi FinPour	l init i c
2	Fin decompte	

Tableau de déclaration des objets locaux

NOM	TYPE	ROLE
l	entier	longueur de la chaîne
init	procédure locale à **decompte**	sert à initialiser à 0 les éléments du tableau TL
i	entier	compteur
c	caractère	compteur

Analyse du module **init**

	DEFPROCinit(**VAR** TL : TLET)	
S	**L.D.E.**	**O.U.**
1	Résultat = TL TL = [ ] Pour c de "A" à "Z" Faire TL[c] ← 0 FinPour	c
2	Fin init	

Tableau de déclaration des objets locaux

NOM	TYPE	ROLE
c	caractère	compteur et en même temps indice

4) Les algorithmes

Algorithme du programme principal **(1pt)**

```
0) Debut nbre_lettres
1) REPETER                                   (Saisie contrôlée   0.5 pt)
        DONNEE ("Chaîne à traiter :")
   JUSQU'A (longueur (ch) dans [1..100])
2) PROCdécompte(ch, LET, NLET)
3) Ecrire(" Votre texte comporte " , longueur(ch), " caractères dont : "),  PROC
   afficher(LET), écrire(" et " , NLET, " caractères non alphabétiques.")
4) Fin nbre_lettres
```

Algorithmes des modules **(2,5pt -0.25 / faute)**

Algorithme du module **afficher** **(1pt)**

```
0) DEFPROCafficher(TL : TLET)
1) Pour c de "A" à "Z" faire
        Si (TL[c] ≠0) alors
            Ecrire(TL[c], " fois la lettre ", c)
        FinSi
   FinPour
2) Fin afficher
```

Algorithme du module **decompte et init** **(1pt)**

```
0) DEFPROCdecompte(cht : chaine[100] ; VAR TL : TLET ; VAR NLT : entier)
1) [l←Long(cht),PROC init(LT)] Pour i de 1 à l Faire
                    [c ← Majus(cht[i])] Si (c dans ["A".."Z"]) alors
                                            TL[c] ← TL[c]+1
                                        Sinon
                                            NLT ← NLT + 1
                                        FinSi
                    FinPour
2) Fin decompte
```

Algorithme du module **init**

```
0) DEFPROC init(VAR TL : TLET)
1) Pour c de "A" à "Z" Faire
```

```
            TL[c] ← 0
        FinPour
2)  Fin init
```

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION
ET DE LA FORMATION

EXAMEN DU BACCALAUREAT

SESSION DE JUIN 2008

ANCIEN REGIME

SECTIONS : SC. EXP. + MATH + TECHNIQUE
EPREUVE : INFORMATIQUE
DUREE : 1h30
COEFFICIENT : 0,5

** Le sujet comporte 2 pages numérotées de 1/2 à 2/2. La réponse à la "PARTIE I" du sujet se fera sur la page 1/2 qui doit être remise à la fin de l'épreuve.*
** La réponse à la "PARTIE II" est à développer sur les feuilles de composition.*

PARTIE I : (6 points)

Exercice 1 (2 points)

Sachant que la fonction RANDOM(n) retourne un entier aléatoire (au hasard) appartenant à l'intervalle [0..n-1], compléter le tableau suivant :

Fonction	Rôle
RANDOM(51)	Retourne, au hasard, un entier appartenant à l'intervalle [0..50]
RANDOM(51) + 10	.....
.....	Retourne, au hasard, un entier appartenant à l'intervalle [10..99]

Exercice 2 (4 points)

Dans un contexte informatique, donner une définition et un exemple pour chaque terme du tableau suivant :

Terme	Définition	Exemple
Protocole	.....	.....
URL	.....	.....
Adresse IP	.....	.....
Adresse mail	.....	.....

PARTIE II (14 points)

Deux entiers sont dits "jumeaux" s'ils sont premiers et impairs successifs.

Un entier est premier s'il n'est divisible que par 1 et par lui-même.

Exemples :

- Les deux entiers 11 et 13 sont jumeaux car ils sont tous les deux premiers et impairs successifs.
- Les deux entiers 29 et 31 sont jumeaux car ils sont tous les deux premiers et impairs successifs.

On se propose d'écrire un programme qui cherche et affiche tous les couples d'entiers jumeaux de l'intervalle [1..100].

Questions

1. Analyser ce problème en le décomposant en modules.
2. Analyser chacun des modules envisagés.
3. En déduire les algorithmes correspondants.

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION
ET DE LA FORMATION

EXAMEN DU BACCALAUREAT

SESSION DE JUIN 2007

SECTIONS :
MATH. + SC. EXP. + TECH.

EPREUVE : INFORMATIQUE

DUREE : 1h30 COEF. : 0,5

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

Nom et prénom : .....

Date et lieu de naissance : .....

Signature des surveillants
.....
.....

Cette feuille est à remettre
à la fin de l'épreuve avec la feuille de copie .

Partie I (6 points)

Exercice 1 *(3 points)*

Dans le tableau ci-dessous, remplir les deux colonnes "Résultat" et " Type du résultat" par le résultat et le type correspondant à chacune des expressions de la première colonne.

Expression	Résultat	Type du résultat
CONCAT (SOUS-CHAINE("Baccalauréat", 1, 3), " 2007")	.....	.....
(("D"<"A") ET (ABS(-1)>0))	.....	.....
(15 DIV 3) MOD 2	.....	.....
TRONC(7.25) + ARRONDI(7.23)	.....	.....

Exercice 2 *(3 points)*

Soit l'algorithme suivant :

0) Début algo
1) Lire(ch,c)
2) [i ← 0, tr ← (1=0)] Répéter
   i ← i + 1
   tr ← (c = ch[i])
   Jusqu'à (tr) ou (i = long(ch))
3) Fin algo

a) Quel est le rôle de cet algorithme ?

.....
.....
.....

b) Compléter le tableau ci-dessous par le type ou la nature des objets qui ont figuré dans l'algorithme précédent.

Objet	Type ou nature
i	.....
c	.....
tr	.....
ch	.....

Partie II *(14 points)*

PHRASE est une chaîne comportant un minimum de 5 caractères. On se propose d'écrire un programme qui saisit PHRASE puis affiche sans répétition :

1) les caractères de PHRASE qui sont des lettres (minuscules ou majuscules, accentués ou non)
2) les autres caractères de PHRASE

Questions :

1) Définir les structures de données à utiliser dans la résolution de ce problème.
2) a) Analyser ce problème
   b) Déduire l'algorithme du programme principal.
3) a) Analyser les modules envisagés dans 2)
   b) Déduire les algorithmes correspondants aux différents modules.

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION ET DE LA FORMATION

EXAMEN DU BACCALAUREAT

SESSION DE JUIN 2006

SECTIONS : MATH. + SC.EXP. + TECH.

EPREUVE : INFORMATIQUE

DUREE : 1h30 COEFFICIENT : 0,5

Cette feuille entière est à remettre à la fin de l'épreuve avec la feuille de copie.

Partie I (6points)

La réponse à cette partie est à développer sur cette même feuille.

Exercice 1 : (3 points)

Soit la fonction booléenne **VERIF** suivante :

```
FUNCTION VERIF (Ch : String) : ..... ;
Var
      ..... ;
      ..... ;
Begin
      Test := False ;
      Vc := 0;
      Repeat
            Vc := Vc + 1;
            If  Not(Upcase(Ch[Vc]) in ['A' .. 'Z']) Then
                              Begin
                                    Test:= True;
                              End;
      Until (Test) Or (Vc = Length(Ch));
      ..... ;
End;
```

Questions :

1) Compléter les pointillés par les données manquantes.
2) Que fait cette fonction ?

.....

.....

Exercice 2 : (3 points)

Mettre devant chaque aperçu le numéro du code **HTML** correspondant.

N°	Code HTML
1	<OL> <LI> Examen pratique </LI> <LI> Examen théorique </LI> </OL>
2	Examen pratique Examen théorique
3	<B> Examen pratique </B> <I> Examen théorique </I>
4	<U> Examen pratique </U> <Center> Examen théorique </Center>
5	Examen pratique Examen théorique

Aperçu	N° du code HTML
Examen pratique Examen théorique	.....
Examen pratique Examen théorique	.....
1. Examen pratique 2. Examen théorique	.....
Examen pratique *Examen théorique*	.....
Examen pratique Examen théorique	.....

Partie II (14 points)

La réponse à cette partie est à développer sur une feuille de copie.

Soient deux tableaux **T1** et **T2** contenant chacun **n** éléments distincts deux à deux (2<**n**<100). On appelle **intersection** de **T1** et **T2** l'ensemble des éléments communs à ces deux tableaux.

On se propose d'écrire un programme qui range les éléments de l'intersection des deux tableaux dans un tableau **INTER** puis affiche les trois tableaux **T1**, **T2** et **INTER.**

Questions :

1) Analyser ce problème en le décomposant en modules.
2) Analyser chacun des modules proposés.
3) Déduire un algorithme du programme principal ainsi que ceux des modules.

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION ET DE LA FORMATION

❖❖❖

EXAMEN DU BACCALAUREAT

❖❖❖

SESSION DE JUIN 2005

SECTIONS : MATH. + SC.EXP. + TECH.

EPREUVE : INFORMATIQUE

DUREE : 1H30 COEFFICIENT : 0,5

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.
La réponse à la "PARTIE I" du sujet se fera sur les pages 1/3 et 2/3 qui doivent être remises à la fin de l'épreuve.

Note

...../ 20

PARTIE I (6 points)

Exercice 1 (3 points)

Soit la fonction **Traitement** suivante écrite en Pascal :

```
FUNCTION Traitement (d,f:integer;T:tab): ..... ;
VAR
      ..... ;
BEGIN
  indmin:=d ;
  FOR i := d+1 TO f DO
   IF T[i] < T[indmin] THEN
       Begin
           indmin := i ;
       End;
  Traitement :=indmin ;
END ;
```

Questions

1/ Déterminer et compléter le type de cette fonction ainsi que la partie déclaration des variables locales.

2/ Quelle est la valeur renvoyée par la fonction **Traitement** si **d=2**, **f=5** et le tableau T contient les éléments suivants :

T	-10	5	0	-6	10	13
i	1	2	3	4	5	6

.....

3/ Quel est le rôle de cette fonction ?

.....

.....

Exercice 2 (3 points)

Dans un contexte informatique, définir les termes suivants :

1) **Signet :** .....

.....

.....

2) **Adresse IP :** .....

.....

.....

3) **Protocole :** .....

.....

.....

PARTIE II (14 points)

Le conseil scientifique d'une institution est formé de m membres avec $10 \leq m \leq 20$ et m impair. Pour décider de l'achat de micro-ordinateurs, les membres du conseil effectuent un vote. Cette opération est informatisée. Chacun des membres exprime son avis par la saisie d'un seul caractère qui peut être :

F ou **f** pour **Favorable**,
D ou **d** pour **Défavorable**,
N ou **n** pour **Neutre**.

On vous demande d'écrire un programme qui affichera la décision à prendre par le conseil sachant qu'elle est :

*"**Reportée**" si le pourcentage des neutres est strictement supérieur à **50 %**, sinon elle est "**Acceptée**" si le pourcentage des favorables est strictement supérieur à celui des défavorables et "**Refusée**" dans le cas contraire.*

N.B : Votre programme devra faire des saisies contrôlées suivant les indications citées précédemment.

Questions

1. Analyser ce problème en le décomposant en modules.
2. Analyser le programme principal ainsi que chacun des modules envisagés.
3. En déduire les algorithmes correspondants.

REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION ET DE LA FORMATION

EXAMEN DU BACCALAUREAT — SESSION PRINCIPALE — JUIN 2005

SECTIONS : MATH. + SC.EX. + TECH. ---- EPREUVE : INFORMATIQUE

PARTIE I (6 points)

Exercice 1 (3 points)

1)

Soit la fonction ***Traitement*** suivante écrite en Pascal :

```
FUNCTION Traitement (d,f:integer;T:tab): INTEGER ;   (Integer 0.5, Real 0.25)
VAR
       indmin     : INTEGER ;   (0.25 point)
       i          : INTEGER ;   (0.25 point)
BEGIN
  indmin:=d ;
  FOR i := d+1 TO f DO
   IF T[i] < T[indmin] THEN
       Begin
              indmin := i ;
       End;
  Traitement :=  indmin ;
END ;
```

2/ La valeur renvoyée par la fonction Traitement si **d=2**, **f=5** et le tableau **T** contient les éléments suivants est :

T	-10	5	0	-6	10	13
i	1	2	3	4	5	6

Avant la boucle, indmin vaut 2

pour i = 3 on a T[3] =0 < 5 = T[2] donc indmin = 3

pour i = 4, on a T[4] = -6 < 0 = T[3] donc indmin = 4

pour i = 5, on a T[5] =10 ≥ -6 = T[4] , indmin reste inchangé

Donc la valeur renvoyée par cette fonction est ***4*** **(1 point)**

3) *Le rôle de cette fonction est de déterminer le premier emplacement du minimum dans la partie de T commençant à l'indice d et se terminant à l'indice f.*

"le premier emplacement" 0.25 point
"le minimum" 0.5 point
"la plage (la partie de T)" 0.25 point

Exercice 2 (3 points)

Dans un contexte informatique, la définition de :

1) Signet : *lieu indiqué dans un document pour référencer un lien hypertexte.*

Mots clés : - Emplacement ou tout mot équivalent (0.5 point)

- Lien hypertexte (0.5 point)

2) Adresse IP : *Identifiant unique d'une machine connectée à un réseau et utilisant le protocole de communication TCP/IP. Une telle adresse et actuellement composée de quatre entiers compris chacun entre 0 et 255.*

Mots clés : - Identifiant ou adresse unique ou toute expression équivalente (0.5 point)

Forme de l'adresse (4 octets) ou adresse d'Internet ou Adresse d'un réseau IP ou TCP/IP (0.5 point)

NB : si le candidat donne seulement un exemple => 0.25 point

3) Protocole : *Ensemble de règles régissant la communication entre machines interconnectées.*

1 point

NB : si le candidat donne seulement un exemple => 0.25 point

PARTIE II (14 points)

1) Le résultat de ce problème est une décision qui dépendra essentiellement du nombre de voix favorables **nbf**, du nombre de voix défavorables **nbd** et du nombre de voix neutres **nbn**. Ces nombres seront le résultat d'une procédure de vote fait par les m membres du conseil. Remarquons que :
nbf + nbd + nbn = m. Il suffit de calculer nbf et nbd et on a donc nbn = m - (nbf + nbd)

Analyse du problème

	NOM : marché_info	
S	**L.D.E.**	**O.U.**
3	**Résultat** = Ecrire (FN décision(nbf, nbd, nbn, m))	nbf, nbd,
2	(nbf, nbd, nbn) = PROC vote(nbf, nbd, nbn, m)	nbn, m
1	m = PROC saisie(m)	décision
		vote
4	**Fin marché_info**	saisie

Codification des objets globaux

OBJET	NATURE/TYPE
nbf	entier
nbp	entier
nbn	entier
m	entier
décision	fonction chaîne
vote	procédure
saisie	procédure

2) Analyse des modules

	DEFFN décision (favorable, defavorable, neutre, n : entier) : chaîne	
S	**L.D.E.**	**O.U.**
1	Résultat = décision décision = [ ] Si (neutre > n **DIV** 2) alors décision ← "Reportée"	

	sinon si (favorable > defavorable) alors décision ← " Acceptée" sinon décision ← "Refusée" Finsi	
2	Fin décision	

	DEFPROC vote (**VAR** favorable, defavorable, neutre : entier ; n : entier)	
S	**L.D.E.**	**O.U.**
	Résultat = (favorable, defavorable, neutre)	
2	neutre = n - (favorable + defavorable)	
1	(favorable, defavorable) = [favorable ← 0, defavorable ← 0]	
	Pour i de 1 à n répéter	i
	Répeter	
	Ecrire ("Vote n° ", i)	voix
	Lire (voix)	
	voix ← MAJUS (voix)	
	Jusqu'à voix dans {"F","D","N"}	
	si (voix ="F") alors	
	favorable ← favorable + 1	
	sinon si (voix = "D") alors	
	defavorable ← defavorable + 1	
	FinSi	
	FinPour	
3	**Fin vote**	

Codification des objets locaux

OBJET	**NATURE/TYPE**
i	entier
voix	caractère

	DEFPROC saisie (**VAR** n : entier)	
S	L.D.E.	O.U.
	Résultat = n	
1	n = [ ] Répéter	
	Ecrire("Donner un entier impair compris entre 10 et 20 ")	
	Lire(n)	valide
	valide ← (n MOD 2 = 1) et (n >= 10) et (n <= 20)	
	jusqu'à valide	
2	Fin saisie	

OBJET	**NATURE/TYPE**
valide	booleen

3) Algorithmes

```
0) Début marché_info
1) PROC saisie (m)
2) PROC vote (nbf, nbd, nbn, m)
3) Ecrire (FN décision(nbf, nbd, nbn, m) )
4) Fin marché_info
```

```
0) DEFPROC saisie (VAR n : entier)
1) Répéter
        Ecrire("Donner un entier impair compris entre 10 et 20 ")
        Lire(n)
        valide ← (n MOD 2 = 1) et (n >= 10) et (n <= 20)
   jusqu'à valide
3) Fin saisie
```

```
0)  DEFPROC vote (VAR  favorable, defavorable, neutre : entier ; n : entier)
1)  [ favorable ← 0, defavorable ← 0]
      Pour i de 1 à n répéter
        Répeter
            Ecrire ("Vote n° ", i)
            Lire (voix)
            voix ←  MAJUS (voix)
          jusqu'à voix dans {"F","D","N"}

        si voix ="F" alors
                    favorable ← favorable + 1
         sinon si voix = "D" alors
                    defavorable ← defavorable + 1
          finsi
      FinPour
2)  neutre ← n - ( favorable + defavorable)
3) Fin vote
```

```
0)  DEFFN décision (favorable, defavorable, neutre, n : entier) : chaine
1)  si (neutre > n DIV 2) alors
                    décision ← "Reportée"
            sinon si (favorable > defavorable ) alors
                    décision ← " Acceptée"
            sinon
                    décision ← "Refusée"
            Finsi
2)  Fin décision
```

Barème

Questions	**Note**
1/ Analyser le problème en le décomposant en modules (3 points)	
- Analyse et décomposition en modules	1 point
- Programme principal	1 point
- Tableaux de déclaration.	0.5 point
- Cohérence des appels et mode de passage	0.5 point
NB : -0.25 point par erreur	
2/ Analyser le programme principal ainsi que chacun des modules envisagés.	
* Module Saisie de m & contrôles (1.5 points):	0.5 point
- Saisie de m	0.25 point
- Boucle Répéter	3x0.25 point
- Contrôle (parité, borne inf, borne sup)	
* Module vote (4 points)	
- Saisie des voix + test	1 point
- Contrôle de la saisie (Répéter … jusqu'à)	0.5 point
- Calcul de nbf, nbd, nbn	3x0.5 points
- Structure répétitive Pour	0.5 point
- Structure conditionnelle Si	0.5 point
* Module décision + affichage (2.5 points)	
décision	3x0.5
Affichage	1 point
3/ En déduire les algorithmes correspondants (3 points)	
moins 0.25 point par erreur.	3 points