

Exercice 1 :

Soit l'algorithme suivant:

```

0) Début fonction Quoi (ch:chaîne; c: caractère) : entier
1) nb ← 0
   pour i de 1 à long(ch) faire
       si ch[i] = c alors
           nb ← nb + 1
       fin si
   fin pour
2) Quoi ← nb
3) Fin Quoi

```

Questions :

1. Exécuter manuellement cette fonction pour ch = " programme " et c = " m "
2. Donner le rôle de la fonction «**Quoi** ».
3. Donner une solution **récursive** de cette fonction.

Réponse

1) pour ch = " programme " et c = " m ".

9	E		
8	M		
7	M		
6	A		
5	R		
4	G		
2	3	0	
1	2	R	
0	1	p	2
NB	i	Ch[i]	quoi



- 2) Rôle : calculer le nombre d'occurrence d'un caractère dans une chaîne.
- 3) La version récursive :

```

0) Début fonction Quoi (ch:chaîne; c: caractère) : entier
1) Si (ch= "") alors
    Quoi ← 0
Sinon si (ch[1]=c) alors
    Efface(ch,1,1)
    Quoi ← 1+ Fn Quoi (ch, c)
Sinon
    Efface(ch,1,1)
    Quoi ← 0+ Fn Quoi (ch, c)
Fin si
2) Fin Quoi

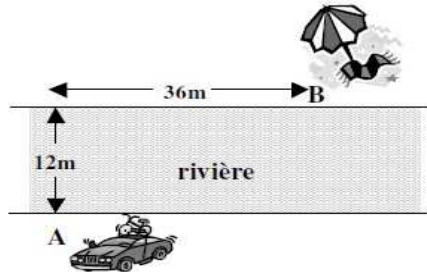
```

Exercice 2 :

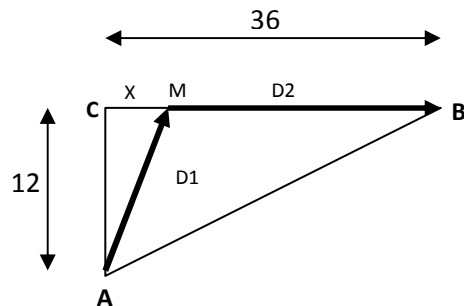
Mohamed veut aller se baigner à la plage Beau-Soleil. Il stationne son auto de l'autre côté de la rivière (point A) et il décide de la traverser pour se rendre à la plage (point B).

Sachant que Mohamed nage à une vitesse de 3 m/s et qu'il marche à une vitesse de 5 m/s, quel trajet permettra à Mohamed de se rendre à la plage (point B) le plus rapidement possible ?

Voici la situation



Voici un schéma de la situation



D = Distance totale

D1 = Distance parcouru dans l'eau

D2 = Distance parcouru au sol

T = Temps total

Rappel :

$$V = D/T$$

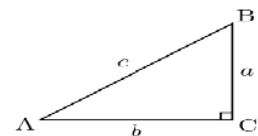
T1 = Temps pour parcourir la distance dans l'eau

T2 = Temps pour parcourir la distance au sol

V1 = Vitesse à la nage : 3 m/s

V2 = Vitesse de marche : 5 m/s

Théorème d'hypoténuse : $c = \sqrt{a^2 + b^2}$



Questions

- 1) Établir une équation reliant les variables entre elles :

$$D = D1 + D2$$

$$D(x) = \dots\dots\dots ?$$

- 2) Déterminer la quantité à optimiser et exprimer cette quantité en fonction d'une seule variable. $T = T1 + T2$

$$T(x) = \dots\dots\dots ?$$

- 3) Ecrire un algorithme d'un module qui reçoit en paramètre le pas de variation de X et retourne la valeur de X, pour laquelle la durée T est minimale.

(On suppose que la fonction **T(x) existe** déjà et bien définie).

Réponse

1) $D(x) = D1(x) + D2(x)$

$$D1(x) = \sqrt{12^2 + x^2}$$

$$D2(x) = 36 - x$$

$$\Rightarrow D(x) = \sqrt{12^2 + x^2} + 36 - x$$

2) $T(x) = T1(x) + T2(x)$

$$T1(x) = D1(x)/V1$$

$$T2(x) = D2(x)/V2$$

$$T(x) = (D1(x)/V1) + (D2(x)/V2)$$

$$T(x) = (\sqrt{12^2 + x^2}/V1) + (36 - x)/V2$$

- 3) Algorithme :

0) Def Fn Calcul(pas :réel) :réel

1) $V1 \leftarrow 3$

2) $V2 \leftarrow 5$

3) $x \leftarrow 0$

4) $t \leftarrow \text{Fn } T(x)$

5) Répéter



```

xp ← x
tp ← t
x ← xp + pas
t ← Fn T(x)
jusqu'à (T > Tp)
6) calcul ← xp
7) Fin Calcul

```

Problème :

Après la réussite au BAC, l'orientation universitaire vise, sur la base du mérite, à garantir une affectation à tout bachelier. En effet, la priorité est à celui qui a le score (nombre de points) le plus élevé. Ce score est appelé formule globale.

Les élèves (bacheliers) admis d'une section donnée seront classés par ordre décroissant selon la formule globale. Puis, une fois classés, ces bacheliers seront divisés en 3 groupes de la façon suivante :

Groupe1 (30%)			Groupe2 (40%)			Groupe3 (30%)		
Rang : 1	..	R1	R1+1	..	R2	R2+1	..	N

Dans le répertoire « c:\Bac2014 », on dispose d'un fichier nommé « SI.dat » contenant la liste des bacheliers admis de la section Sciences de l'informatique.

Dans ce fichier, chaque bachelier est défini par :

- **Num_insc** : le numéro d'inscription (chaîne de 6 chiffres).
- **NP** : Le nom et prénom (chaîne de 40 caractères au maximum).
- **MG** : moyenne générale.
- **FS** (formule spécifique) : un réel déjà calculé à partir des notes obtenues dans les diverses matières.
- **i** : un réel = 1 si l'élève est redoublant en BAC et 1.05 sinon.
- **B** : un réel compris entre 0 et 5 représentant une bonification sur le rang au cours de l'année.

On souhaite réaliser un programme qui permet de :

1. Créer un autre fichier « SI_FG.dat », à partir du fichier « SI.dat », et y stocker, pour les mêmes bacheliers, les informations suivantes :

- **Num_insc** : le numéro d'inscription (chaîne de 6 chiffres)
- **NP** : Le nom et prénom (chaîne de 40 caractères au maximum)
- **FG** : Formule Globale (réel)

Sachant que La formule globale (FG) de chaque élève est calculée par l'équation :

$$FG = ((5 * MG + FS) * i) + B$$

2. Classer les bacheliers du fichier « SI_FG.dat » par ordre décroissant selon la formule globale (FG).

N.B : Pour faire cette tâche, on doit transmettre les informations dans un tableau, les trier puis les transmettre ordonnées dans le fichier.

3. Extraire, dans le même répertoire, 3 autres fichiers (« SI_g1.dat », « SI_g2.dat » et « SI_g3.dat ») contenant respectivement les bacheliers appartenant au groupe 1, groupe 2 et groupe 3.

4. Afficher, pour un candidat donné, en fonction de son numéro d'inscription (donnée à saisir), le groupe auquel il appartient.

Travail demandé :

- ➡ Analyser le problème en le décomposant en modules.
- ➡ Analyser chaque module envisagé.
- ➡ En déduire les algorithmes correspondants.

Problème :

Analyse

Nom : Orientation

Résultat= Ecrire("le candidat appartient au groupe", Fn Chercher(ins, g1,g2, g3))

```

Ins=[ ] Proc Lecture( ins )
(g1,g2,g3)=[ ] Proc orienter( ffg, g1, g2, g3 )
(Ffg,n)=[ ] Proc Trier( ffg )
Ffg=[ ] Proc Extraire( Fsi, Ffg )
(Ffg,Fsi,g1,g2,g3)= Proc Association( Fsi, Ffg, g1, g2, g3 )
Fin Orientation
  
```

T.D.N.T

Type
candidat=enregistrement
Num_insc : chaîne[6]
np : chaîne[40]
MG ,FS : réel
i :réel {1 ou 1.05}
B : réel {0 ou 5}
Fin candidat
Cand_reduit=enregistrement
Num_insc_r : chaîne[6]
Np_r : chaîne[40]
FG : réel
Fin Cand_reduit
F1 = Fichier de candidat
F2 = Fichier de cand_reduit
Tab=tableau de 500 cand_reduit

T.D.O.G

Objet	Type/Nature
Lecture	Procédure
orienter	Procédure
Trier	Procédure
trier	Procédure
extraire	Procédure
Association	Procédure
Chercher	Fonction
Fsi	F1
Ffg,g1,g2,g3	F2
Ins	Chaîne
Cand	Candidat
Cand_r	Cand_reduit
t	tab

Analyse de la procédure Association

DEF Proc Association (Var Fsi :F1 ;var Ffg,g1,g2,g3 :F2)

Résultat = (Fsi,Ffg,g1,g2,g3)

```

G3= ReCréer(g3),
g3= Associer (g3, " C:\ bac2014\si_g3.dat ")
G2= ReCréer(g2)
g2= Associer (g2, " C:\ bac2014\si_g2.dat ")
G1= ReCréer(g1)
g1= Associer (g1, " C:\ bac2014\si_g1.dat ")
ffg = Recréer(ffg)
ffg= Associer (ffg, " C:\ bac2014\si_fg.dat ")
fsi = Associer (fsi, "C:\ bac2014\si.dat")
  
```

FIN Association

Analyse de la procédure Extraire

```

DEF Proc Extraire (var Fsi :F1 ;var Ffg :f2)
Résultat = Ffg
ffg = [ Ouvrir(fsi) ] tantque non(fin fichier(fsi)) faire
    Lire(fsi, cand)
    Avec cand faire
        cand_r.Num_insc_r ← Num_insc
        cand_r.Np_r ← Np
        cand_r.FG ← ((5*mg+fs)* i)+b
    Fin avec
    Ecrire(ffg, cand_r)
Fin tant que
FIN Extraire
{à la fin fermer(fsi) et fermer(ffg)}

```

Analyse de la procédure Trier

```

DEF Proc Trier (var ffg :f2 )
Résultat = ffg
Ffg= [ ] Proc Importer ( t, n, ffg )
T [ ] = Pour i de 2 à n faire
    [J<i ,Aux←t[i] ] Tant que ( j-1>0) et (t[j-1].FG >aux.FG ) Faire
        T [ j ]←-- T [ j-1 ]
        j <--j - 1
    Fin Tant Que
    T[j]←aux
Fin pour
(t, n) = [ ] Proc Exporter ( ffg, t, n )
FIN Trier
T.D.O.L

```

Objet	Type/Nature
i,j	Entier
aux	Cand_reduit

Analyse de la procédure Exporter

```

DEF Proc Exporter (Var ffg : F2 ; var t : tab ;var i :entier)
Résultat = ( t,i )
(t,i) = [ ouvrir (ffg), i←0 ] tant que( non (fin_fichier(ffg)) )faire
    i←i+1
    lire( ffg, t[i] )
fin tantque
FIN Exporter
{à la fin Fermer(ffg)}
T.D.O.L

```

Objet	Type/Nature
i	Entier

Analyse de la procédure Importer

```

DEF Proc Importer (t :tab ; n :entier ; var ffg :f2 )
Résultat = ffg
Ffg = [ ReCréer(Ffg) ] Pour i de 1 à n faire
    Ecrire(ffg, t[i])

```

```

    fin pour
{à la fin Fermer(ffg)}
FIN Importer
T.D.O.L

```

Objet	Type/Nature
I	entier

Analyse de la procédure Orienter

```

DEF Proc Orienter ( Var ffg,g1,g2,g3 : F2 )
Résultat = ( g1,g2,g3 )
(g1,g2,g3) = [ i←0 ] tant que( non (fin_fichier(ffg)) )faire
    i←i+1
    lire( ffg, cand_r )
    selon i faire
        1.. r1 : écrire(g1,cand_r)
        R1+1 .. r2 : écrire(g2,cand_r)
        Sinon écrire(g3,cand_r)
    Fin selon
fin tantque
r2← r1+ arrondi(n*40/100)
r1← arrondi(n*30/100)
n ← Taille_fichier(ffg)
ouvrir (ffg)
FIN Orienter
{à la fin Fermer(ffg) , Fermer(g1) , Fermer(g2) et Fermer(g3) }
T.D.O.L

```

Objet	Type/Nature
I,n,r1,r2	Entier

Analyse de la procédure Lecture

```

DEF Proc Lecture (Var ins :chaine )
Résultat = ins
ins = [ ] Répéter
    ins=donnée("Saisir le numéro d'inscription (6 chiffres) ! ")
    valeur(ins, nb_ins, e)
    jusqu'à ( long(ins)=6 ET e=0 )
FIN Lecture
T.D.O.L

```

Objet	Type/Nature
Nb_ins, e	Entier

Analyse de la fonction Chercher

```

DEF Fn Chercher ( ins :chaine ; Var g1,g2,g3 : F2 ) :chaine
Résultat = Chercher
Chercher = Chercher ←g
g = [ ] Si ( Fn Groupe( ins ,g1) ) alors
    g←"Rose"
    sinon Si ( Fn Groupe( ins ,g2 ) ) alors
        g←"Vert"
    Sinon si ( Fn Groupe( ins , g3 ) ) alors

```

```

                g ← "Blanc"
            sinon
                g ← "n'existe pas"
        Fin si
FIN Chercher

```

Analyse de la fonction groupe

```

DEF Fn groupe ( ins :chaîne ; g :f2 ) :booléen
Résultat = Groupe
Groupe = Groupe ← exit
exit = [ exit ← faux, ouvrir(g) ] repeter
    lire(g, cand_r)
    Si(cand_r.Num_insc_r = ins ) alors
        Exist ← vrai
    Fin si
Jusqu'à (exist=vrai) ou (fin_fichier(g) )
FIN groupe
{à la fin Fermer(g)}
T.D.O.L

```

Objet	Type/Nature
exit	booléen

Algorithme du programme principal

- 0) **Debut** Orientation
- 1) Proc Association(Fsi, Ffg, g1, g2, g3)
- 2) Proc Extraire(Fsi,Ffg)
- 3) Proc Trier(ffg)
- 4) Proc orienter(ffg, g1 ,g2 ,g3)
- 5) Proc Lecture(ins)
- 6) Ecrire("le candidat appartient au groupe", Fn Chercher(ins, g1, g2, g3))
- 7) **Fin** Orientation

Algorithme de la procédure association

- 0) **DEF Proc** Association (Var Fsi :F1 ;var Ffg,g1,g2,g3 :F2)
- 1) Associer (fsi, "C:\ bac2014\si.dat")
- 2) Associer (ffg, " C:\ bac2014\si_fg.dat ")
- 3) Recréer(ffg)
- 4) Associer (g1, " C:\ bac2014\si_g1.dat ")
- 5) ReCréer(g1)
- 6) Associer (g2, " C:\ bac2014\si_g2.dat ")
- 7) ReCréer(g2)
- 8) Associer (g3, " C:\ bac2014\si_g3.dat ")
- 9) ReCréer(g3)
- 10) **FIN** Association

Algorithme de la procédure Extraire

- 0) **DEF Proc** Extraire (var Fsi :F1 ;var Ffg :f2)
- 1) Ouvrir(fsi)
- 2) tantque non(fin fichier(fsi)) faire
 - Lire(fsi, cand)
 - Avec cand faire
 - cand_r.Num_insc_r ← Num_insc

```

        cand_r.Np_r ← Np
        cand_r.FG ← ((5*mg+fs)* i)+b
    Fin avec
    Ecrire(ffg, cand_r)
Fin tant que
3) fermer(fsi)
4) fermer(ffg)
5) FIN Extraire

```

Algorithme de la procédure Exporter

```

0) DEF Proc Exporter (Var ffg : F2 ; var t : tab ;var i :entier)
1) ouvrir (ffg)
2) i←0
3) tant que( non (fin_fichier(ffg)) )faire
    i←i+1
    lire( ffg, t[i] )
fin tantque
4) Fermer(ffg)
5) FIN Exporter

```

Algorithme de la procédure Importer

```

0) DEF Proc Importer (t :tab ; n :entier ; var ffg :f2 )
1) Recréer(Ffg) { Ouvrir le fichier et effacer son contenu }
2) Pour i de 1 à n faire
    Ecrire(ffg, t[i])
fin pour
3) Fermer(ffg)
4) FIN Importer

```

Algorithme de la procédure Trier

```

0) DEF Proc Trier (var ffg :f2 )
1) Proc Exporter ( ffg, t ,n )
2) Pour i de 2 à n faire
    J←i
    Aux←t[i]
    Tant que ( j-1>0) et (t[j-1].FG >aux.FG ) Faire
        T [ j ]←-- T [ j-1 ]
        j <--j - 1
    Fin Tant Que
    T[j]←aux
fin pour
3) Proc Importer (t ,n , ffg )
4) FIN Trier

```

Algorithme de la procédure Orienter

```

0) DEF Proc Orienter ( Var ffg,g1,g2,g3 : F2 )
1) ouvrir (ffg)
2) N← Taille_fichier( ffg )
3) r1← arrondi(n*30/100)
4) r2← r1+ arrondi(n*40/100)
5) i←0
6) tant que( non (fin_fichier(ffg)) )faire
    i←i+1

```



```

lire( ffg, cand_r )
selon i faire
    1.. r1 : ecrire(g1,cand_r)
    R1+1 .. r2 : ecrire(g2 ,cand_r)
    Sinon ecrire(g3 ,cand_r)
Fin selon
fin tantque
7) Fermer(ffg)
8) Fermer(g1)
9) Fermer(g2)
10) Fermer(g3)
11) FIN Orienter

```

Algorithme de la procédure lecture

```

0) DEF Proc Lecture (Var ins :chaine )
1) Répéter
    Ecrire("Saisir le numéro d'inscription (6 chiffres) ! ")
    Lire(ins)
    Valeur (ins, nb_ins, e )
    jusqu'à ( long(ins)=6  ET  e=0 )
2) FIN Lecture

```

Algorithme de la fonction Groupe

```

0) DEF Fn groupe ( ins :chaine ; g :f2 ) :booléen
1) exit ← faux
2) ouvrir(g)
3) repeter
    lire(g, cand_r)
    Si( cand_r .Num_insc_r = ins ) alors
        Exist ← vrai
    Fin si
    Jusqu'à (exist=vrai) ou (fin_fichier(g) )
4) Fermer(g)
5) FIN groupe

```



Algorithme de la fonction chercher

```

0) DEF Fn Chercher ( ins :chaine ; Var g1,g2,g3 :f2 ) :chaine
1) Si ( Fn Groupe( ins ,g1) ) alors
    g ← "Rose"
    sinon Si ( Fn Groupe( ins ,g2 ) ) alors
        g ← "Vert"
        Sinon si ( Fn Groupe( ins ,g3 ) ) alors
            g ← "Blanc"
        sinon
            g ← "n'existe pas"
    Fin si
2) Chercher ← g
3) FIN Chercher

```