Baccalauréat 2008
Algorithmique et programmation
Solution du sujet de la session de contrôle

Exercice 1 : (4 pt)

1°) Les variables a et b sont de type booléen ou logique. (0.75 + 0.75)

2°) Exécution à la main (-0.25 / faute). (2.5 pt)

- 0.5 pt pour l'affichage et le formatage : on se contentera de quelques espaces réguliers pour supposer que l'élève a compris le formatage
- On notera la table de vérité indépendamment du traitement des boucles
- Si les valeurs logiques sont mises entre « » ou en minuscules ne pas pénaliser

a	**b**	**a OR b**	**a AND b**	0.5 pt
TRUE	TRUE	TRUE	TRUE	
TRUE	FALSE	TRUE	FALSE	
FALSE	TRUE	TRUE	FALSE	
FALSE	FALSE	FALSE	FALSE	
0.25 pt	0.25 pt	0.5 pt	0.5 pt	

Exercice 2 : (4 pt)

a) Une fonction est dite récursive lorsque **dans sa définition, elle fait appel à elle-même**. (1 pt)

b) Une méthode de calcul du PGCD de p et q et qui est par définition récursive est celle de la différence et elle est définie ainsi : (1 pt)

PGCD (p , q) = PGCD (p-q , q) si p > q
ou = PGCD (p , q-p) si p < q

On continue ce procédé jusqu'à l'arrivée à la situation de calcul de PGCD (k,k), dans ce cas, le PGCD sera égale à k.

Exemple : PGCD (18,24)=PGCD (18,6)=PGCD (12,6)=PGCD (6 ,6)=6

- On pourra penser à la méthode euclidienne.
- Nom de la méthode (0.5 pt) allusion au traitement récursif.(0.5 pt)
- Si en guise de réponse, le candidat donne un exemple illustré (0.5 pt)

c) Analyse de la fonction PGCD ou équivalent quand la solution est juste. (1 pt)

- (-0.25 / faute)

```
DEFFN PGCD (p,q : entier) : entier
Résultat =  PGCD
1   PGCD = [     ] Si p = q Alors
                      PGCD ← p
          Sinon  Si p > q  Alors
                              PGCD ← PGCD (p-q , q)
                       Sinon PGCD ← PGCD (p, q-p)
          Finsi
```

2 Fin

d) **Algorithme de la fonction PGCD** (1 pt)

- (-0.25 / faute)

```
0) DEF FN PGCD (p,q : entier) : entier
1) Si (p=q) Alors
            PGCD ←p
   Sinon  Si (p>q)  Alors
                  PGCD ← PGCD(p-q , q)
            Sinon PGCD ← PGCD(p , q-p)
    Finsi
2) Fin  PGCD
```

Problème :

1°) **Analyse du programme principal**

- **le formalisme de la solution n'est pas important, on évaluera toute forme de solution.**

Résultat = agences.dat
5 agences.dat = PROC Fusionner ($ag\_sfax_{trié}$, ag_meden, agences)
4 agences = associer (agences,"c:\agences.dat")
3 $ag\_sfax_{trié}$ = PROC Tri_file (ag_sfax)
2 ag_sfax = associer (ag_sfax , "c:\ag_sfax.dat")
1 ag_mednine = associer(ag_meden, "c:\ag_meden.dat")
6 Fin

Analyse (7 pts)

Traitement à évaluer	Détails	Barème
P.P.	Modularité + cohérence (1 + 0.5)	1.5
Création agence	Fusion (interne ou externe) Tri (interne à la mémoire) Transfert	1.5 1.5 1
T.D.O.		0.5
Association	**On acceptera une seule association**	0.5
Ouverture / Fermeture		0.5

2°) **Analyse de la procédure Fusionner**

```
DEFPROC Fusionner  (var ag_sfax , ag_meden, agences : Tfile)
  Résultat : Agences
  Agences = (Partie1 , Partie2)
5  Partie1 = [ recréer(agences) ; i ← 1 ; j←1] Tant que (i ≤ cs) et (j≤cm) faire
                                          [    ] Si  T_sfax[i].nom ≤ T_medn[j].nom alors
                                                  Ecrire (agences,T_sfax[i])
                                                  i:=i+1
                                              Sinon
                                                  Ecrire (agences,T_medn[j])
                                                  j:=j+1
                                               Finsi
                                          Fin Tant que
```

```
6  Partie2 = [    ] Si  ( i > cs) alors
            [   ] Pour k de j à cm faire
                        Ecrire (agences,T_medn[k]) ;
                  Fin Pour
            Sinon
            [   ] Pour k de i à cs faire
                        Ecrire (agences,T_sfax[k])
                  Fin Pour
            Fin si
7  Fermer (agences)

1  (T_sfax , cs) = [ouvrir(ag_sfax) ; cs←0]  Tant que Non (fin fichier(ag_sfax)) faire
                                              Lire(ag_sfax, f )
                                              cs←cs+1
                                              T_sfax[cs] ←f
                                        Fin Tant que
2  Fermer(ag_sfax)
3  (T_medn , cm) = [ouvrir(ag_meden)  ; cm←0] Tant que Non (fin fichier(ag_meden)) faire
                                                    Lire(ag_meden,f)
                                                    cm←cm+1
                                                    T_medn[cm] ←f
                                              Fin Tant que
4  fermer(ag_meden)
8  Fin
```

Analyse de la procédure tri_file

```
   DEFPROC Tri_file (var ag_sfax : Tfile)
   Résultat : Ag_sfax_trié
4  Ag_sfax_trié=[ recréer(ag_sfax)] Pour i de 1 à n faire
                                        Ecrire(ag_sfax,T_trié [i])
                                  Fin Pour
5  Fermer(ag_sfax)
3  T_trié = Proc Tri (n,T)
1  (T  , n) =  [ ouvrir(ag_sfax)  ;  n← 0 ] Tant que non(fin fichier(ag_sfax)) faire
                                              Lire(ag_sfax , f)
                                              n ← n + 1
                                              T[n] ← f
                                        Fin Tant que
2  Fermer(ag_sfax)
6  Fin
```

Analyse de la procédure tri

```
  DEFPROC Tri (n : entier ; var T : Vect)
  Résultat : T trié
1  T trié= [     ] Répéter
                [Permut ← faux , i ←0] Pour i de 1 à n-1 faire
                                                Si T[i] .nom> T[i+1]. nom  Alors
                                                                Aux ← T[i]
                                                                T[i] ← T[i+1]
                                                                T[i+1] ← Aux
                                                                Permut ← vrai
                                                Fin Si
                                        Fin Pour
                n ← n-1
            Jusqu'à (permut = faux)
  Fin
```

3°) Les algorithmes

Algorithmes (5 pts)

Traitement à évaluer	Détails	Barème
P.P.	Modularité + cohérence	1
Création agence	Fusion (interne ou externe) Tri (interne à la mémoire) Transfert	1 1 1
Association	**On acceptera une seule association**	0.5
Ouverture / Fermeture		0.5

Algorithme du programme principal

0) Début PP
1) associer(ag_meden, "c:\ag_meden.dat"
2) associer(ag_sfax , "c:\ag_sfax.dat")
3) PROC Tri_file (ag_sfax)
4) associer (agences,"c:\agences.dat")
5) PROC Fusionner ($ag\_sfax_{trié}$, ag_meden, agences)
6) Fin

Tableau de déclaration des nouveaux types globaux

Type
Fonct = enregistrement Num_mat : entier Nom : chaîne[40] Prenom : chaîne[40] Heures_travail : entier Fin Fonct Tfile = fichier de fonct

Tableau de déclaration des objets globaux

Objet	Type/Nature	Rôle
Ag_meden	Tfile	Fichier de l'agence de Médenine
Ag_sfax	Tfile	Fichier de l'agence de Sfax
Agences	Tfile	Fichier qui va regrouper les deux précédents
Tri_file	Procédure	Procédure de tri des éléments du fichier de Sfax
Fusionner	Procédure	Procédure de fusion des éléments des 2 fichiers

Algorithme de la procédure Fusionner

```
0) Début Proc Fusionner (ag_sfax ,ag_meden : Tfile ; VAR agences : Tfile)
1) [ouvrir(ag_sfax) ;cs←0]Tant que Non (fin fichier(ag_sfax)) faire
                                Lire(ag_sfax,f)
                                cs←cs+1
                                T_sfax[cs] ←f
                            Fin Tant que
   Fermer(ag_sfax)
2) [ouvrir(ag_meden) ; cm←0] Tant que Non (fin fichier(ag_meden)) faire
                                Lire(ag_med,f)
                                cm←cm+1
                                T_medn[cm] ←f
                            Fin Tant que
   fermer(ag_meden)
3) [recréer(agences) ; i ← 1 ; j←1] Tant que (i ≤ cs) et (j≤cm) faire
                                Si T_sfax[i].nom ≤ T_medn[j].nom alors
                                    Ecrire(agences,T_sfax[i])
                                    i:=i+1
                                Sinon
                                    Ecrire(agences,T_medn[j])
                                    j:=j+1
                                Finsi
                            Fin Tant que
4) Si ( i > cs) alors
                    Pour k de j à cm faire
                        Ecrire (agences,T_medn[k]) ;
                    Fin Pour
      Sinon
            Pour k de i à cs faire
                    Ecrire (agences,T_sfax[k])
            Fin Pour
      Fin si
   Fermer(agences)
5) Fin Fusionner
```

Tableau de déclaration des objets de la procédure Fusionner

Objet	Type/Nature	Rôle
cs	entier	Compteur pour remplir T_sfax
T_sfax	tableau de 20 fonct	Tableau pour stocker les enregistrements contenus dans le fichier ag_sfax
cm	entier	Compteur pour remplir T_medn
T_medn	tableau de 15 fonct	Tableau pour stocker les enregistrements contenus dans le fichier ag_meden

i	entier	Compteur utilisé pour fusionner les deux tableaux
j	entier	Compteur utilisé pour fusionner les deux tableaux
k	entier	Compteur utilisé pour fusionner les deux tableaux

Algorithme de la procédure Tri_file

0) DEFPROC Tri_file (var ag_sfax : Tfile)
1) [Ouvrir(ag_sfax) ; n ← 0]Tant que non(fin fichier(ag_sfax)) faire
   Lire(ag_sfax , f)
   n ← n + 1
   T[n] ← f
   Fin Tant que
   Fermer(ag_sfax)
2) Proc Tri (n,T)
3) [recréer(ag_sfax)] Pour i de 1 à n faire
   Ecrire(ag_sfax,$T_{trié}$ [i])
   Fin Pour
   Fermer(ag_sfax)
4) Fin Tri_file

Tableau de déclaration des nouveaux types

Type
Vect = tableau de Fonct 15 éléments

- On acceptera la déclaration d'un seul tableau, au lieu de la déclaration de 3 tableaux (T-Sfax(15), T-Medn (20), Vect (35))

Tableau de déclaration des objets

Objet	Type/Nature	Rôle
n	entier	Nombre d'enregistrements contenus dans le fichier ag_sfax
T	Vect	Tableau pour stocker les enregistrements contenus dans le fichier ag_sfax
i	entier	Compteur

Algorithme de la procédure Tri

0) DEFPROC Tri (n : entier ; var T : Vect)
1) Répéter
   Permut ← faux
   i ←0
   Répéter
   i ← i + 1
   Si T[i] .nom> T[i+1]. Nom Alors
   Aux ← T[i]
   T[i] ← T[i+1]
   T[i+1] ← Aux
   Permut ← vrai
   Fin Si
   Jusqu'à i=n-1
   n ← n-1
   Jusqu'à permut = faux
2) Fin Tri

Tableau de déclaration des objets

Objet	Type/Nature	Rôle
permut	booléen	Variable pour vérifier s'il a eu une permutation
Aux	Fonct	Variable auxiliaire pour permuter le contenu de deux cases
i	entier	Compteur