ANNEXE 1 page 1 **CORRIGÉ** **23 juin 1999**

```
--------------------------  SRI_B importe les sémaphores et les opérations P, V et E0
with SRI_B; use SRI_B; with Table_Generique; use Table_Generique;
package body Entrepot is ------------------------------------------------------ ENTREPOT
   N1 : constant Positive := 50; N2 : constant Positive := 5000;
   N : constant := N1 + N2;                          -- total des places dans l'entrepôt
   subtype Index1 is Positive range 1..N1; subtype Index2 is Positive range 1..N2;
-----------------  données communes partagées par les procédures du paquetage
   Stock1 : array (Index1) of Document; Stock2 : array (Index2) of Document;
   package Table1 is new Table_Generique (taille => N1, Element => Natural);
   package Table2 is new Table_Generique (taille => N2, Element => Natural);
   Numero : Natural := 0;          -- sert ànuméroter les clés de façon unique jusqu' à Integer'Last
   Mutex_Numero, Mutex1, Mutex2, S_Stock, S_Stock2: Semaphore;
                          -- en italique la réponse pour la question 6 et annexe 5
-------------------------------------------------------------------------------- ENTREPOT
   procedure Deposer (D : in Document; Ma_Cle : out Cle) is
      K1 : Table1.Index; K2 : Table2.Index; Succes : boolean;
   begin
      P(Mutex_Numero);  -- acquisition d'une cle unique pour ce nouveau document entreposé
      Numero := Numero + 1; Ma_Cle := Numero;
      V(Mutex_Numero);
      P(S_Stock);       -- Ya-t-il assez de place dans Table1 ou Table2 et Stock1 ou Stock2 ?
      P(Mutex1);
      Table1.Rechercher(K1, 0, Succes);       -- recherche une place K1 telle que  Table1(K1) = 0
      if Succes then                                    -- la recherche est  réussie
         Table1.Changer(K1, Ma_Cle);          -- on note l'occupation dans Table1 avec Ma_Cle
         Stock1(K1) := D;                            -- on écrit dans stock1 et c'est fini
         V(Mutex1);
         return;                                     -- sortie de la procedure Deposer
      end if;
      V(Mutex1);
      P(S_Stock2);                -- pour la question 6  voir annexe 5
      P(Mutex2);                  -- on n'a pas trouvé dans Table1, on recherche dans Table2
      Table2.Rechercher(K2, 0, Succes);    -- recherche une place K2 telle que Table2(K2) = 0 ;
      Table2.Changer(K2, Ma_Cle );         -- on est sûr de trouver une place dans Table2
      Stock2(K2) := D;                                --on écrit dans Stock2 et c'est fini
      V(Mutex2);
   end Deposer ;
-------------------------------------------------------------------------------- ENTREPOT
   procedure Consulter (D : out Document; Ma_Cle : in Cle) is
      K1 : Table1.Index; K2 : Table2.Index; Succes : boolean;
   begin
      P(Mutex1);
      Table1.Rechercher(K1, Ma_Cle, Succes); -- recherche K1 tel que Table1(K1) = Ma_Cle
      if Succes then                              -- on a trouvé, la recherche est Succes
         D := Stock1(K1);                                 --on lit dans stock1 et c'est fini
         V(Mutex1);
         return;                                  -- sortie de la procedure Consulter
      end if;
      V(Mutex1);
      P(Mutex2);
                                 -- on n'a pas trouvé dans Table1, on cherche dans Table2
      Table2.Rechercher(K2, Ma_Cle, Succes);   -- on est sûr de trouver Table2(K2) = Ma_Cle
      D := Stock2(K2);                                 --on lit dans stock2 et c'est fini
      V(Mutex2);
   end Consulter;
```

ANNEXE 1 page 2

CORRIGÉ

```
---------------------------------------------------------------------------------- ENTREPOT
   procedure Supprimer (Ma_Cle : in Cle) is
      K1 : Table1.Index; K2 : Table2.Index; Succes : boolean;
   begin
      P(Mutex1);
      Table1.Rechercher(K1, Ma_Cle, Succes);  -- recherche K1 tel que Table1(K1) = Ma_Cle
      if Succes then                                    -- la recherche est un Succes
         Table1.Changer(K1, 0);                          -- on libère la place en inscrivant 0
         V(Mutex1); V(S_Stock);
         return;                                    -- sortie de la procedure Supprimer
      end if;
      V(Mutex1);
      P(Mutex2);
      Table2.Rechercher(K2, Ma_Cle, Succes);          --  on n'a pas trouvé dans Table1
      Table2.Changer(K2, 0);  -- on est sûr de trouver Table2(K2) = Ma_Cle et on libère la place
      V(Mutex2);
      V(S_Stock);    V(S_Stock2);                     -- on compte une place libre de plus
   end Supprimer;
---------------------------------------------------------------------------------- ENTREPOT
   procedure Modifier (D : in Document; Ma_Cle : in Cle) is
      K1 : Table1.Index; K2 : Table2.Index; Succes : boolean;
   begin
      P(Mutex1);
      Table1.Rechercher(K1, Ma_Cle, Succes);-- recherche K1 tel que Table1(K1) = Ma_Cle
      if Succes then                                          -- la recherche est Succes
         Stock1(K1) := D;                                  --on écrit D dans stock1 et c'est fini
         V(Mutex1);
         return;                                    -- sortie de la procedure Modifier
      end if;
      V(Mutex1);
      P(Mutex2);
      Table2.Rechercher(K2, Ma_Cle, Succes);   -- on est sûr de trouver Table2(K2) = Ma_Cle
      Stock2(K2) := D;                                  --on écrit D dans stock2 et c'est fini
      V(Mutex2);
   end Modifier;
------------------
begin
   E0(Mutex_Numero, 1); E0(Mutex1, 1); E0(Mutex2, 1); E0(S_Stock, N);
   E0(S_Stock2, N2);
            --initialisation des sémaphores
end Entrepot;          ----------------------------------------------------------------ENTREPOT
```

ANNEXE 2

CORRIGÉ

```
with SRI_B; use SRI_B; with Entrepot; use Entrepot;
package body Transaction is ------------------------------------------ TRANSACTION
S_Fifo, Mutex_Consultation, S_Depot, S_ModSupp : semaphore;
NC : Natural := 0;
-------------------
   procedure Suite_Consultations (Nb : in positive; S : in Liste_Cle) is
      Doc : Document;
   begin
      P(S_Fifo);
      P(Mutex_Consultation);
      NC := NC + 1;
      if NC = 1 then P(S_ModSupp); end if;
      V(Mutex_Consultation);
      V(S_Fifo);
      for I in 1..Nb loop
         Consulter (Doc, S(I));              -- consultation de Nb documents
      end loop;
      P(Mutex_Consultation);
      NC := NC - 1;
      if NC = 0 then V(S_ModSupp); end if;
      V(Mutex_Consultation);
   end Suite_Consultations;
------------------------------------------------------------------ TRANSACTION
   procedure Suite_avec_Modification_Suppression (Nb : in positive; S : in Liste_Cle) is
   begin
      P(S_Fifo);
      P(S_ModSupp);
      V(S_Fifo);
      for I in 1..Nb loop
         Consulter(Doc, S(I));          - - consultation de Nb documents
         Modifier(Doc, S(I));             -- modification du document consulte
      end loop;
      for I in 2..Nb-1 loop
         Supprimer(Doc, S(I));              -- suppression de document
      end loop;
      V(S_ModSupp);
   end Suite_avec_Modification_Suppression;
------------------------------------------------------------------ TRANSACTION
   procedure Creation (D : in Document; Ma_Cle : out Cle) is
   begin
      P(S_Depot);
      Déposer (D, Ma_Cle);
      V(S_Depot);
   end Creation;
------------------------------------------------------------------ TRANSACTION
begin                                  -- initialisation des sémaphores etc...
   E0(S_Fifo, 1);
   E0(Mutex_Consultation, 1); E0(S_ModSupp, 1);
   E0(S_Depot, 5);
end Transaction;------------------------------------------------- TRANSACTION
```

ANNEXE 5

CORRIGÉ

```
---------------------------------------------------------------- DÉMON DE L'ENTREPOT
-- le démon déplace le plus ancien document de Table1 vers Table2 si on peut, sinon on ne fait
rien
   task Demon;
   task body Demon is
      Minuit : duration := 0.1;                    --simule le réveil à minuit
      Plus_Petite_Cle : Positive := 1;             -- il n'y a pas de clé plus petite dans Table1
      K1 : Table1.Index; K2 : Table2.Index; Succes;
   begin
   loop
      delay Minuit;                                -- lancer le Démon de minuit

      P(S_Stock2);
      -- évite de prendre la dernière place de Table2 qui pourrait être déjà réservée par
      -- un dépot concurrent avec le démon
      P(Mutex1);
      if Table1.Place < N1/2 then                  -- il n'y a pas trop de places vides dans Stock1
         while not Succes loop
            Table1.Rechercher(K1, Plus_Petite_Cle , Succes);   -- La clé est-elle encore présente?
            Plus_Petite_Cle := Plus_Petite_Cle + 1;            -- sinon on recherche la clé suivante
            end loop;                                          -- la recherche a-t-elle abouti?
         P(Mutex2);
         Table2.Rechercher(K2, 0, Succes);         -- on recherche une place libre, c'est K2
         if Succes then                                        -- Table2 n'est pas pleine
            Table2.Changer(K2, Mon_Numero);
            Stock2(K2) := Stock1(K1);                  -- on ne déplace que si K2 existe
            Table1.Changer(K1, 0);
         else                                                  -- Table2 est pleine
            Plus_Petite_Cle := Plus_Petite_Cle - 1;    -- on doit corriger pour la fois suivante
         end if;
         V(Mutex2);
      else
         V(S_Stock2); -- s'il y a trop de places vides, on ne déplace pas de document  cette fois
      end if;
      V(Mutex1);

   end loop;
   end Demon; ------------------------------------------------- DÉMON DE L'ENTREPOT
```