```
generic
    type T_Element is private;  -- Type des éléments de la pile

package Piles is

    type T_Pile is private;

     Pile_Vide:Exception;
                    --TESTS--

        --Initilaiser une pile.  La pile est vide.
    procedure Initialiser (Pile : out T_Pile) with
        Post => Est_Vide (Pile);
        --Vérifier si la pile est vide.
    function Est_Vide (Pile : in T_Pile) return Boolean;
        --Vérifier si un élément existe dans la pile.
    function Existe_Pile(Element: in T_Element; Pile: in T_Pile) return Boolean;


            --FONCTIONS ELEMENTAIRES--

        --Récupérer l'élément en sommet de la pile.
    function Sommet (Pile : in T_Pile) return T_Element;
        --Récupérer la cellule suivante de la pile.
    function Next_Pile(Pile: in T_Pile) return T_Pile;
        --Récupérer la taille d'une pile
    function Size_Pile(Pile: in T_Pile) return Integer;


              --AJOUT/SUPPRESSION--

        --Empiler l'élément en somment de la pile.
    procedure Empiler (Pile : in out T_Pile; Element : in T_Element) with
        Post => Sommet (Pile) = Element;
        --Supprimer l'élément en sommet de pile
    procedure Depiler (Pile : in out T_Pile) with
        Pre => not Est_Vide (Pile);
        --Supprimer un élément de la pile
    procedure Supprimer_Element(Element: in T_Element; Pile: in out T_Pile);
        -- Détruire la pile.
    procedure Detruire (P: in out T_Pile);
        --Affecter la première pile par la deuxième
    procedure Affecter_Pile(P:in out T_Pile;P2: in T_Pile);


                    --AFFICHAGE--

    -- Afficher les éléments de la pile
    generic
        with procedure Afficher_Element (Un_Element: in T_Element);
    procedure Afficher_Pile (Pile : in T_Pile);


private

    type T_Cellule;

    type T_Pile is access T_Cellule;

    type T_Cellule is
        record
            Element: T_Element;
            Suivant: T_Pile;
        end record;

end Piles;
```