```
with Piles;

Generic
    TYPE T_Value is PRIVATE; --Donnée d'une clé.
    Zero: in T_Value;        --Le "zéro" du type T_Value. (Sera -
34404 ou 'E' pour signifier ERROR quand la fonction devra retourner quelque ch
ose même s'il n'y a rien à retourner)

package Arbre_Binaire is
        TYPE T_Node is LIMITED PRIVATE; --
Les cellules d'un arbre, i.e. les noeuds, d'où le nom "node" en anglais.
        TYPE T_Branch is PRIVATE; --
Le pointeur qui accèdera aux fils d'un noeud, d'où le nom "branche".

        package Piles_Cle is
            new Piles (T_Element => Integer);
        use Piles_Cle; --Module de piles qui sera très utile plus tard.

                                --EXCEPTIONS--

    Cle_Presente_Exception : Exception; -
- une clé est déjà présente dans un ABR
    Cle_Absente_Exception  : Exception; -- une clé est absente d'un ABR
    Arbre_Vide             : Exception;


                        --FONCTIONS/PROCEDURES SECONDAIRES--

        --Procédure obligatoire pour instancier le module des piles.
    procedure Afficher_Entier(Element: in Integer);


                                --TESTS--

        --Initialiser un arbre vide.
    procedure Initialiser_Vide(Arbre: out T_Branch);
        --Initialiser un arbre avec  la clé de sa racine éventuelle.
    procedure Initialiser (Cle: in Integer; Arbre : out T_Branch);
        -- Est-ce que le noeud s'agit d'une feuille? Est-
ce que l'abre est vide?
    function Est_Nul (Arbre : in T_Branch) return Boolean;
        --
Savoir si une clé est la racine ou bien le fils droit ou le fils gauche de son
 ancêtre. (retourne 'R' pour racine, 'D' pour fils droit et 'G' pour fils gauc
he)
    function Gauche_ou_Droite(Cle:in Integer;Arbre: in T_Branch) return Charac
ter;
```

```
                    --FONCTIONS ELEMENTAIRES--

        --Retourner la clé du noeud.
    function Nodekey(Arbre: in T_Branch) return Integer;
        --Retourner la valeur du noeud.
    function NodeValue(Arbre:in T_Branch) return T_Value;
        --Retourner le fils droit du noeud.
    function Fils_Droit(Arbre: in T_Branch) return T_Branch;
        --Retourner le fils gauche du noeud.
    function Fils_Gauche(Arbre: in T_Branch)return T_Branch;
        --Multiplier toutes les clés de l'arbre par 10.
    procedure Multiplier_10(Arbre: in out T_Branch);
        --Retourner la profondeur d'un arbre.
    function Depth(Arbre: in T_Branch) return Integer;
        --Affecter un noeud par un autre
    procedure Affecter_Arbre(Arbre1: in out T_Branch; Arbre2: in T_Branch);



                --FONCTIONS/PROCEDURES GENERATIONNELLES-

        --Récupérer la génération d'un noeud par rapport à la racine.
    function Gen(Cle: in Integer; Arbre:in T_Branch) return Integer;
        --Récupérer le nombre de fils d'un noeud.
    function Nbr_Fils_Noeud(Cle: in Integer; Arbre: in T_Branch) return Intege
r;
        --Récupérer l'ensemble des fils d'un noeud
    function Ensemble_Fils_Noeud(Cle: IN Integer;Arbre: in T_Branch) return Pi
les_Cle.T_Pile;
        --Récupérer le nombre de noeuds de génération g.
    function Nbr_Meme_Generation(g: in integer; Arbre: in T_Branch) return Int
eger;
        --
Afficher l'ensemble des noeuds de génération g par rapport à un noeud.
    procedure Ensemble_Meme_Generation(g,Cle:in Integer; Arbre: in T_Branch);
        --
Afficher l'ensemble des noeuds de génération g ou moins par rapport à un noeud
.
    procedure Ensemble_n_Generation(g,Cle:in Integer; Arbre: in T_Branch);
        --Récupérer l'ensemble des noeuds ayant un seul fils.
    function Ensemble_Un_Fils(Arbre:in T_Branch) return Piles_Cle.T_Pile;
        --Récupérer l'ensemble des noeuds ayant deux fils.
    function Ensemble_Deux_Fils(Arbre:in T_Branch) return Piles_Cle.T_Pile;
        --Récupérer l'ensemble des feuilles d'un arbre.
    function Ensemble_Feuilles(Arbre:in T_Branch) return Piles_Cle.T_Pile;



                --FONCTIONS/PROCEDURES DE RECHERCHE--
```

```
        --Récupérer la cellule d'un noeud à partir de sa clé.
    function Rech_Noeud(Cle: in Integer; Arbre: in T_Branch) return T_Branch;
        --Récupérer la cellule de l'ancêtre d'un noeud à partir de sa clé.
    function Rech_Ancetre(Cle: in Integer; Arbre: in T_Branch) return T_Branch
;
        --
Affecter à Noeud la cellule de clé Cle dans l'arbre (i.e., affecter le resulta
t de Rech_Noeud(Cle,Arbre) à Noeud).
    procedure Affecter_Rech_Noeud(Cle: in Integer;Arbre: in T_Branch; Noeud: i
n out T_Branch);
        --Récuperer la donnée associée à une clé dans un arbre.
    function Donnee_Noeud(Cle : in Integer ; Arbre : in T_Branch) return T_Val
ue;
        --Récupérer la clé associée à une clé dans un arbre.
    function Cle_Noeud(Cle:in Integer; Arbre:in T_Branch) return Integer;



                                --AJOUT--

        --
Récupérer l'intervalle (piles) de valeurs permises que peut prendre le fils év
entuel d'un noeud en testant avec une clé quelconque.
        --par convention, la pile sera de la forme [max,0,-
181199] pour représenter l'intervalle ]-INFINI,max], [-
181199,0,min] pour représenter l'intervalle [min,+INFINI[ et [max,min] pour re
présenter l'intervalle [min,max].
    function NewKeyInterval(Cle,Parent: Integer; Arbre:T_Branch) return Piles_
Cle.T_Pile;
        --
Insérer un nouvel élément (clé + donnée) dans l'arbre binaire de façon automat
ique.
    procedure Inserer(Cle: in Integer; Donnee: in T_Value; Arbre: in out T_Bra
nch);
        --Insérer un nouvel élément (clé + donnée) sous un noeud précis.
    procedure Ajouter2(Cle_Nouveau_Noeud: in Integer; Donnee_Nouveau_Noeud: in
 T_Value; Cle_Noeud_Parent:in integer; Arbre: in out T_Branch);



                            --MODIFICATION--

        --Modifier la clé de la racine ou du noeud poussé en argument.
    procedure Modifier_Cle_Racine(NewCle:in Integer;Arbre: in out T_Branch);

        --Modifier la clé d'un noeud.
    procedure Modifier_Cle(Cle,NewCle: in Integer; Arbre: in out T_Branch);
        --Modifier la donnée d'un noeud.
```

```
    procedure Modifier_Donnee(Cle: in Integer; NewDonnee: in T_Value; Arbre: i
n out T_Branch);


                              --SUPPRESSION--

        -- Supprimer les fils d'un noeud de l'arbre.
    procedure Supprimer_Fils (Arbre: in out T_Branch);
        -- Supprimer un noeud de l'arbre ainsi que ses fils.
    procedure Supprimer_Cle_ET_Fils(Cle: in Integer; Arbre: in out T_Branch);
        -- Détruire tout un arbre ou un noeud et tous ses fils.
    procedure Detruire(Arbre: in out T_Branch);


                              --AFFICHAGE--

    generic
        with procedure Afficher_Donnee (Donnee: in T_Value);
    procedure Afficher_ABR (Arbre : in T_Branch); --Afficher l'arbre complet.
    generic
        with procedure Afficher_Donnee (Donnee: in T_Value);
    procedure Afficher_APartir(Cle:in Integer; Arbre: in T_Branch); --
Afficher un arbre à partir d'une clé.




PRIVATE
    TYPE T_Branch is ACCESS T_Node; --Pointeur qui accèdera aux noeuds.
    TYPE T_Node is
        RECORD
            Cle: Integer;           --La clé d'un noeud.
            Donnee: T_Value;        --La donnée d'une clé.
            FilsG: T_Branch;        --
Pointeur qui accède au fils gauche d'un noeud.
            FilsD: T_Branch;        --
Pointeur qui accède au fils droit d'un noeud.
        end RECORD;
end Arbre_Binaire;
```