```
with Ada.Text_IO;                   use Ada.Text_IO;
with Ada.Integer_Text_IO;           use Ada.Integer_Text_IO;
with Ada.Strings.Unbounded;         use Ada.Strings.Unbounded;
with Ada.Text_IO.Unbounded_IO;      use Ada.Text_IO.Unbounded_IO;
with Arbre_Genealogique;            use Arbre_Genealogique;
with Registre;                      use Registre;

procedure main is
    package Piles_AG renames Arbre_Genealogique.Arbre_Binaire_Character.Piles_Cle;
    package ABR renames Arbre_Genealogique.Arbre_Binaire_Character;
    AG,NoeudAncetre:ABR.T_Branch;
    RG:T_Access;
    Quitter:Boolean:=False;
    MenuPrecedent: Boolean:=False;
    Commencer:Character:='N';
    Addinfo:Character:='N';
    Racine:Integer;
    NbrFils_Avant,NbrFils_Apres,g,NewMonth_Integer,NewDay,NewYear,New_Key,Newer_Key,New_Key
_Desc,Option,Option1,Option2,Option3:Integer;
    New_Donnee:Character;
    New_Name,NewBirthP:Unbounded_String;
    NewMonth:Registre.T_Mois;

    procedure Start(Cle:in Integer;AG:in out ABR.T_Branch;RG: in out T_Access) is
        begin
            Start_RG(Cle,RG);
            Init_AG(Cle,AG);
        end Start;

    function Sont_Vides(AG: in ABR.T_Branch; RG: in T_Access) return Boolean is
        begin
            if Est_Vide_RG(RG) and Est_Nul_AG(AG) then
                return True;
            elsif (not Est_Vide_RG(RG)) and Est_Nul_AG(AG) then
                Put_Line("Seul l'arbre est vide!");
                return False;
            elsif Est_Vide_RG(RG) and not Est_Nul_AG(AG) then
                Put_Line("Seul le registre est vide!");
                return False;
            else
                Put_Line("Ni l'arbre ni le registre est vide.");
                return False;
            end if;
        end Sont_Vides;


    function Existe(Cle: in Integer;AG: in out ABR.T_Branch; RG: in T_Access) return Boolea
n is
```

```
        begin
            if Existe_RG(Cle,RG) and Est_Present(Cle,AG) then
                return True;
            elsif (not Existe_RG(Cle,RG)) and Est_Present(Cle,AG) then
                Put_Line("La clé existe dans l'arbre mais pas dans le registre!");
                return False;
            elsif Existe_RG(Cle,RG) and not Est_Present(Cle,AG) then
                Put_Line("La clé existe dans le registre mais pas dans l'arbre!");
                return False;
            else
                Put_Line("La clé n'existe nul part!");
                return False;
            end if;
        end Existe;

    procedure Edit_Key(Cle,NewCle: in Integer; AG: in out ABR.T_Branch; RG: in out T_Access
) is
        CleAncetre,NewParentKey,FGKey,FDKey,TMPKEY,GAUCHEOUDROIT:Integer;
        Noeud: ABR.T_Branch;
        Ens:Piles_AG.T_Pile;
        NewKey:Integer:=NewCle;
        AjoutPossible: Boolean:=False;
        choix: Character:='n';
        begin
            if (not Est_Nul_AG(AG)) and (not Est_Vide_RG(RG)) then
                if Cle=NewCle then
                    Null;
                elsif NewCle=-181199 or NewCle=-34404 then
                    Put_Line("Veuillez saisir une autre clé, les clés -34404 et -
181199 sont utilisées intérieurement par ce programme pour assurer son fonctionnement..");
                elsif not ABR.Est_Nul(Rech_Noeud_AG(NewCle,AG)) then --
Si NewCle existe déjà  dans l'arbre
                    Put_Line("Existe déjà !");
                else
                    if (not ABR.Est_Nul(Rech_Noeud_AG(Cle,AG))) then
                        if not ABR.Est_Nul(ABR.Rech_Ancetre(Cle,AG)) then
                            CleAncetre:=ABR.Nodekey(ABR.Rech_Ancetre(Cle,AG));
                            if ABR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(CleAncetre,AG)))=C
le then
                                GAUCHEOUDROIT:=CleAncetre-1;
                            else
                                GAUCHEOUDROIT:=CleAncetre+1;
                            end if;
                            Ens:=New_Key_Interval(GAUCHEOUDROIT,CleAncetre,AG);
                            if not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG))) then
                                FDKey:=ABR.Nodekey(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG)))-
1;
                            end if;
```

```
                            if not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle,AG))) then
                                FGKey:=ABR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(Cle,AG)))+
1;
                            end if;
                            if Piles_AG.Sommet(Ens)=-
181199 and Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and (not Piles_AG.Est_Vide(Piles_AG.N
ext_Pile(Piles_AG.Next_Pile(Ens)))) and (not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG
)))) then -- intervalle du type [min,+inf[
                                Piles_AG.Depiler(Ens);Piles_AG.Depiler(Ens);
                                if (not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle,AG))))
 and FGKey>Piles_AG.Sommet(Ens) then
                                    Piles_AG.Depiler(Ens);Piles_AG.Empiler(Ens,FGKey);Piles
_AG.Empiler(Ens,FDKey);
                                else
                                    Piles_AG.Empiler(Ens,FDKey);
                                end if;
                            elsif ((not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle,AG))))
 and (Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and (not Piles_AG.Est_Vide(Piles_AG.Next_P
ile(Piles_AG.Next_Pile(Ens)))))) and then Piles_AG.Sommet(Piles_AG.Next_Pile(Piles_AG.Next_
Pile(Ens)))=-181199  then --intervalle du type ]-inf,max]
                                if (not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG))))
and FDKey<Piles_AG.Sommet(Ens)then
                                    Piles_AG.Depiler(Ens);Piles_AG.Depiler(Ens);Piles_AG.De
piler(Ens);Piles_AG.Empiler(Ens,FGKey);Piles_AG.Empiler(Ens,FDKey);
                                else
                                    TMPKEY:=Piles_AG.Sommet(Ens);
                                    Piles_AG.Depiler(Ens);Piles_AG.Depiler(Ens);Piles_AG.De
piler(Ens);Piles_AG.Empiler(Ens,FGKey);Piles_AG.Empiler(Ens,TMPKEY);
                                end if;
                            elsif Piles_AG.Est_Vide(Piles_AG.Next_Pile(Piles_AG.Next_Pile(E
ns))) and Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))<Piles_AG.Sommet(Ens) and ((not Est_Nul_A
G(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG)))) or (not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(
Cle,AG))))) THEN --intervalle du type [min,max] avec min<max
                                if (not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle,AG))))
and FDKey<Piles_AG.Sommet(Ens) then
                                    Piles_AG.Depiler(Ens);Piles_AG.Empiler(Ens,FDKey);
                                end if;
                                if (not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle,AG))))
 and FGKey>Piles_AG.Sommet(Piles_AG.Next_Pile(Ens)) then
                                    TMPKEY:=Piles_AG.Sommet(Ens);Piles_AG.Depiler(Ens);Pile
s_AG.Depiler(Ens);
                                    Piles_AG.Empiler(Ens,FGKey);Piles_AG.Empiler(Ens,TMPKEY
);
                                end if;
                            end if;
                            if Piles_AG.Sommet(Ens)=-
181199 and Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and not Piles_AG.Est_Vide(Piles_AG.Ne
xt_Pile(Piles_AG.Next_Pile(Ens))) then -- intervalle du type [min,+inf[
```

```
                                while NewKey < Piles_AG.Sommet(Piles_AG.Next_Pile(Piles_AG.
Next_Pile(Ens))) or (NewKey=-181199 or NewKey=-34404 ) loop
                                    Put("Valeur invalide! Doit être supérieur à  " & Intege
r'Image(Piles_AG.Sommet(Piles_AG.Next_Pile(Piles_AG.Next_Pile(Ens)))) & ". Saisissez une au
tre valeur : ");
                                    Get(NewKey);
                                    New_Line;
                                end loop;
                                AjoutPossible:=True;
                            elsif (Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and (not Pile
s_AG.Est_Vide(Piles_AG.Next_Pile(Piles_AG.Next_Pile(Ens))))) and then Piles_AG.Sommet(Piles
_AG.Next_Pile(Piles_AG.Next_Pile(Ens)))=-181199 then --intervalle du type ]-inf,max]
                                while Piles_AG.Sommet(Ens)<NewKey or (NewKey=-
181199 or NewKey=-34404 ) loop
                                    Put("Valeur invalide! Doit être inférieur à  " & Intege
r'Image(Piles_AG.Sommet(Ens)) & ". Saisissez une autre valeur : ");
                                    Get(NewKey);
                                    New_Line;
                                end loop;
                                AjoutPossible:=True;
                            elsif Piles_AG.Est_Vide(Piles_AG.Next_Pile(Piles_AG.Next_Pile(E
ns))) and Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))<Piles_AG.Sommet(Ens) THEN --
intervalle du type [min,max] avec min<max
                                while NewKey < Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))  or
 Piles_AG.Sommet(Ens) < NewKey or (NewKey=-181199 or NewKey=-34404 )loop
                                    Put("Valeur invalide! Doit être entre " & Integer'Image
(Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))) & " et" & Integer'Image(Piles_AG.Sommet(Ens)) &
". Saisissez une autre valeur : ");
                                    Get(NewKey);
                                    New_Line;
                                end loop;
                                AjoutPossible:=True;
                            elsif Piles_AG.Est_Vide(Piles_AG.Next_Pile(Piles_AG.Next_Pile(E
ns))) and  Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=Piles_AG.Sommet(Ens) THEN --
une seule valeur possible!
                                while NewKey /= Piles_AG.Sommet(Ens) loop
                                    Put("Valeur invalide! Doit être égale à " & Integer'Ima
ge(Piles_AG.Sommet(Ens)) & ". Saisissez cette valeur : ");
                                    Get(NewKey);
                                    New_Line;
                                end loop;
                                AjoutPossible:=True;
                            else --aucune valeur possible!
                                Put_Line("Modification impossible! Il va falloir modifier l
a clé: " & Integer'Image(CleAncetre) & " (clé prédécesseur).");
                                Put("Voulez-
vous la modifier? [y/n] :"); Get(choix); New_Line;
                                if choix='n' or choix='N' then
```

```
                                    Null;
                                else
                                    Put("Saisissez la valeur de la nouvelle clé que vous vo
ulez attribuer à  " & Integer'Image(CleAncetre) & " : "); Get(NewParentKey); New_Line;
                                    while NewParentKey=-181199 or NewParentKey=-34404 loop
                                        Put("Veuillez saisir une autre clé, les clés -
34404 et -
181199 sont utilisées intérieurement par ce programme pour assurer son fonctionnement : ");
Get(NewParentKey);
                                        New_Line;
                                    end loop;
                                    Edit_Key(CleAncetre, NewParentKey, AG,RG);
                                    Put_Line("Vous avez modifié la clé du prédécesseur.");
                                    Put_Line("Vous allez maintenant ressayer de modifier la
 première clé (" & Integer'Image(NewCle) & " ).");
                                    Edit_Key(Cle,NewCle,AG,RG);
                                    AjoutPossible:=False;
                                end if;
                            end if;
                            if AjoutPossible then
                                Affecter_Rech_Noeud_AG(Cle,AG,Noeud);
                                Modifier_Cle_Racine_AG(NewKey,Noeud);
                                ModifyKey(Cle,NewKey,RG);
                            else
                                Null;
                            end if;
                        else --
Si la clé existe mais n'a pas d'ancêtre (i.e., c'est la racine)
                            Modifier_Cle_Racine_AG(NewKey,AG);
                            ModifyKey(Cle,NewKey,RG);
                        end if;
                    else
                        Put_Line("Inexistante!");
                    end if;
                end if;
            else
                Start(NewCle,AG,RG);
            end if;
        end Edit_Key;

    procedure Add_Ancestor(Cle_Nouveau_Noeud: in out Integer; Donnee_Nouveau_Noeud: in Char
acter; Cle_Noeud_Parent:in integer; AG: in out ABR.T_Branch; RG: in out T_Access) is
        Ens:Piles_AG.T_Pile;
        NewKey:Integer:=Cle_Nouveau_Noeud;
        AjoutPossible: Boolean:=False;
        choix: Integer;
        NewParentKey:Integer;
        leftright:Character;
```

```
        Grand_Ancestor:Integer;
        begin
            if (Est_Nul_AG(AG)) and (Est_Vide_RG(RG)) then
                Start(Cle_Noeud_Parent, AG, RG);
                Ajouter_Ancetre(Cle_Nouveau_Noeud,Donnee_Nouveau_Noeud,Cle_Noeud_Parent,AG)
;
                AddKey(Cle_Nouveau_Noeud, RG);
            elsif Cle_Nouveau_Noeud=-181199 or Cle_Nouveau_Noeud=-34404 then
                Put_Line("Veuillez saisir une autre clé, les clés -34404 et -
181199 sont utilisées intérieurement par ce programme pour assurer son fonctionnement..");
            elsif ABR.Est_Nul(ABR.Rech_Noeud(Cle_Noeud_Parent,AG)) then --
Si le prédécesseur n'existe pas
                Put_Line("Il faut d'abord créer le prédécesseur!");
            elsif not Est_Nul_AG(Rech_Noeud_AG(Cle_Nouveau_Noeud,AG)) then --
Si l'arbre n'est pas vide et que le prédécesseur existe mais que la nouvelle clé existe déj
à
                Put_Line("Existe déjà!");
            elsif (not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle_Noeud_Parent,AG)))) and
(not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle_Noeud_Parent,AG)))) then
                Put_Line("Plus de place!");
            elsif Cle_Nouveau_Noeud>Cle_Noeud_Parent and not Est_Nul_AG(ABR.Fils_Droit(Rech
_Noeud_AG(Cle_Noeud_Parent,AG))) then
                if Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(Cle_Noeud_Parent,AG))) then
                    Put_Line("Emplacement rempli! Ressayez avec une clé inférieure à celle
du descendant.");
                else
                    Put_Line("Plus de place! Tentez plutôt une modification...");
                end if;
            elsif Cle_Nouveau_Noeud<Cle_Noeud_Parent and not Est_Nul_AG(ABR.Fils_Gauche(Rec
h_Noeud_AG(Cle_Noeud_Parent,AG))) then
                if Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(Cle_Noeud_Parent,AG))) then
                    Put_Line("Emplacement rempli! Ressayez avec une clé supérieure à celle
du descendant.");
                else
                    Put_Line("Plus de place! Tentez plutôt une modification...");
                end if;
            else
                 Ens:=New_Key_Interval(Cle_Nouveau_Noeud,Cle_Noeud_Parent,AG);
                 if Piles_AG.Sommet(Ens)=-
181199 and Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and not Piles_AG.Est_Vide(Piles_AG.Ne
xt_Pile(Piles_AG.Next_Pile(Ens))) then
                    while NewKey < Piles_AG.Sommet(Piles_AG.Next_Pile(Piles_AG.Next_Pile(En
s))) or (NewKey=-181199 or NewKey=-34404 ) loop
                        Put("Valeur invalide! Doit être supérieur à " & Integer'Image(Piles
_AG.Sommet(Piles_AG.Next_Pile(Piles_AG.Next_Pile(Ens)))) & ". Saisissez une autre valeur :
");
                        Get(NewKey);
                        New_Line;
```

```
                    end loop;
                    AjoutPossible:=True;
                elsif (Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=0 and (not Piles_AG.Est_Vid
e(Piles_AG.Next_Pile(Piles_AG.Next_Pile(Ens))))) and then Piles_AG.Sommet(Piles_AG.Next_Pil
e(Piles_AG.Next_Pile(Ens)))=-181199 then
                    while Piles_AG.Sommet(Ens)<NewKey or (NewKey=-181199 or NewKey=-
34404 ) loop
                        Put("Valeur invalide! Doit être inférieur à " & Integer'Image(Piles
_AG.Sommet(Ens)) & ". Saisissez une autre valeur : ");
                        Get(NewKey);
                        New_Line;
                    end loop;
                    AjoutPossible:=True;
                elsif Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))<Piles_AG.Sommet(Ens) THEN
                    while NewKey < Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))  or Piles_AG.So
mmet(Ens) < NewKey or (NewKey=-181199 or NewKey=-34404 ) loop
                        Put("Valeur invalide! Doit être entre " & Integer'Image(Piles_AG.So
mmet(Piles_AG.Next_Pile(Ens))) & " et" & Integer'Image(Piles_AG.Sommet(Ens)) & ". Saisissez
 une autre valeur : ");
                        Get(NewKey);
                        New_Line;
                    end loop;
                    AjoutPossible:=True;
                elsif Piles_AG.Sommet(Piles_AG.Next_Pile(Ens))=Piles_AG.Sommet(Ens) THEN
                    while NewKey /= Piles_AG.Sommet(Ens) loop
                        Put("Valeur invalide! Doit être égale à " & Integer'Image(Piles_AG.
Sommet(Ens)) & ". Saisissez cette valeur : ");
                        Get(NewKey);
                        New_Line;
                    end loop;
                    AjoutPossible:=True;
                else
                    Put_Line("Insertion impossible! Il va falloir modifier la clé: " & Inte
ger'Image(Cle_Noeud_Parent) & " (clé prédécesseur) ou multiplier toutes les clés de l'arbre
 par 10.");
                    Put_Line("1. Modifier la clé" & Integer'Image(Cle_Noeud_Parent));
                    Put_Line("2. Multiplier toutes les clés de l'arbre par 10");
                    Put("Choisissez une option: "); Get(choix); New_Line;
                    case choix is
                        when 1=>
                            Put("Saisissez la valeur de la nouvelle clé que vous voulez att
ribuer à " & Integer'Image(Cle_Noeud_Parent) & " : "); Get(NewParentKey); New_Line;
                            leftright:=ABR.Gauche_ou_Droite(Cle_Noeud_Parent,AG);
                            if leftright/='R' then
                                Grand_Ancestor:=ABR.Nodekey(ABR.Rech_Ancetre(Cle_Noeud_Pare
nt,AG));
                            end if;
                            while NewParentKey=-181199 or NewParentKey=-34404 loop
```

```
                                Put("Veuillez saisir une autre clé, les clés -34404 et -
181199 sont utilisées intérieurement par ce programme pour assurer son fonctionnement : ");
Get(NewParentKey);
                                New_Line;
                            end loop;
                            Edit_Key(Cle_Noeud_Parent, NewParentKey, AG, RG);
                            Put_Line("Vous allez maintenant ressayer d'ajouter la première
clé (" & Integer'Image(NewKey) & " ).");
                            if leftright='G' then
                                NewParentKey:=ABR.Nodekey(ABR.Fils_Gauche(ABR.Rech_Noeud(Gr
and_Ancestor,AG)));
                            elsif leftright='D' then
                                NewParentKey:=ABR.Nodekey(ABR.Fils_Droit(ABR.Rech_Noeud(Gra
nd_Ancestor,AG)));
                            else
                                NewParentKey:=ABR.Nodekey(AG);
                            end if;
                            Add_Ancestor(NewKey,Donnee_Nouveau_Noeud,NewParentKey,AG,RG);
                        when 2=>
                            ABR.Multiplier_10(AG);
                            RG_Multiplier_10(RG);
                            if Cle_Nouveau_Noeud>Cle_Noeud_Parent then
                                Put_Line("La clé" & Integer'Image(NewKey) & " que vous voul
iez ajouter à la clé" & Integer'Image(Cle_Noeud_Parent*10) & " est maintenant devenue" & In
teger'Image(Cle_Noeud_Parent*10 +5) & ".");
                                Put_Line("Essai d'ajouter" & Integer'Image(Cle_Noeud_Parent
*10 +5) & " à la clé" & Integer'Image(Cle_Noeud_Parent*10) & " :");
                                Cle_Nouveau_Noeud:=Cle_Noeud_Parent*10 +5;
                                Add_Ancestor(Cle_Nouveau_Noeud,Donnee_Nouveau_Noeud,Cle_Noe
ud_Parent*10,AG,RG);
                            else
                                Put_Line("La clé" & Integer'Image(NewKey) & " que vous voul
iez ajouter à la clé" & Integer'Image(Cle_Noeud_Parent*10) & " est maintenant devenue" & In
teger'Image(Cle_Noeud_Parent*10 -5) & ".");
                                Put_Line("Essai d'ajouter" & Integer'Image(Cle_Noeud_Parent
*10 -5) & " à la clé" & Integer'Image(Cle_Noeud_Parent*10) & " :");
                                Cle_Nouveau_Noeud:=Cle_Noeud_Parent*10 -5;
                                Add_Ancestor(Cle_Nouveau_Noeud,Donnee_Nouveau_Noeud,Cle_Noe
ud_Parent*10,AG,RG);
                            end if;
                        when others=> Put_Line("Option saisie invalide!");
                    end case;
                end if;
                if AjoutPossible then
                    Ajouter_Ancetre(NewKey,Donnee_Nouveau_Noeud,Cle_Noeud_Parent,AG);
                    AddKey(NewKey,RG);
                    Cle_Nouveau_Noeud:=NewKey;
                else
```

```
                    Null;
                end if;
            end if;
        end Add_Ancestor;

    procedure Edit_Sexe(Cle: in Integer; Value: in Character; AG: in out ABR.T_Branch) is
        begin
            Modifier_Sexe_AG(Cle,Value,AG);
        end Edit_Sexe;

    procedure Add_Name(Cle: in Integer; Nom: in Unbounded_String; RG:in out T_Access) is
        begin
            AddName(Cle,Nom,RG);
        end Add_Name;

    procedure Add_BirthD(Cle: in Integer; Jour: in Integer; Mois: in T_Mois; Annee:in Integ
er; RG: in out T_Access) is
        begin
            AddBirthD(Cle,Jour,Mois,Annee,RG);
        end Add_BirthD;


    procedure Add_BirthP(Cle: in Integer; Lieu: in Unbounded_String; RG: in out T_Access) i
s
        begin
            AddBirthP(Cle,Lieu,RG);
        end Add_BirthP;

    function Full_Name(Cle: in Integer; RG: in T_Access) return Unbounded_String is
        begin
            return Name(Cle,RG);
        end Full_Name;

    function Birth_Date(Cle: in Integer; RG: in T_Access) return T_Date is
        begin
            return BirthD(Cle,RG);
        end Birth_Date;

    function Birth_Year(Cle: in Integer; RG: in T_Access) return Integer is
        begin
            return BirthY(Cle,RG);
        end Birth_Year;

    function Birth_Place(Cle: in Integer; RG: in T_Access) return Unbounded_String is
        begin
            return BirthP(Cle,RG);
        end Birth_Place;
```

```
    procedure Delete(Cle: in Integer; AG: in out ABR.T_Branch; RG: in out T_Access) is
        Ens: Piles_AG.T_Pile;
        begin
            if Existe(Cle,AG,RG) then
                Ens:=ABR.Ensemble_Fils_Noeud(Cle,AG); --Les ancêtres de la clé Cle
                while not Piles_AG.Est_Vide(Ens) loop --
Tant que la pile des des ancêtres de Cle n'est pas vide
                    Delete_RG(Piles_AG.Sommet(Ens),RG); --
Supprimer un des ancêtres de la clé du registre
                    Piles_AG.Depiler(Ens);              --Dépiler la pile
                end loop; --
La boucle finit quand tous les ancêtres de Cle ont été supprimés du registre
                --
Le traitement précédent qui supprime tous les ancêtres d'une clé du registre ne figure pas
dans le module Registre parce qu'il a besoin de l'arbre généalogique pour reconnaître les a
ncêtres de cette clé
                Delete_RG(Cle,RG);                      --Supprimer la clé Cle du registre
                Supprimer_Famille(Cle,AG);              --
Supprimer la clé Cle et tous ses ancêtres de l'arbre
            else
                Null;
            end if;
        end Delete;

    function Ancestor_Nbr(Cle: in Integer; AG: in ABR.T_Branch) return Integer is
        begin
            return Nombre_Ancetres(Cle,AG);
        end Ancestor_Nbr;

    procedure Same_Gen_Keys(g,Cle:in Integer; AG: in ABR.T_Branch) is
        begin
            Ensemble_Ancetres_Meme_Generation(g,Cle,AG);
        end Same_Gen_Keys;

    procedure Same_Gen_Orless(g,Cle: in Integer; AG: in ABR.T_Branch) is
        begin
            Ensemble_Ancetres_Generation_N(g,Cle,AG);
        end Same_Gen_Orless;

    procedure Print_From(Cle: in Integer; AG: in ABR.T_Branch) is
        begin
            Afficher_A_Partir(Cle,AG);
        end Print_From;

    procedure Print(AG: in ABR.T_Branch) is
        begin
            Afficher_AG(AG);
        end Print;
```

```
    function Homonymes(m1,m2: in Integer; AG: in ABR.T_Branch; RG: in T_Access) return Bool
ean is
        Node1:constant ABR.T_Branch:=Rech_Noeud_AG(m1,AG); --Noeud de m1
        Node2:constant ABR.T_Branch:=Rech_Noeud_AG(m2,AG); --Noeud de m2
        B1,B2:Boolean :=False;                              -
- Booléens qui vont déterminer si un homonyme a été trouvé
        procedure Intermediaire1(Parcours1,Parcours2: in ABR.T_Branch; AG: in ABR.T_Branch;
 RG: in T_Access) is
            begin
                if not Est_Nul_AG(Parcours2) then
                    if Full_Name(ABR.Nodekey(Parcours2),RG)/=To_Unbounded_String("") and th
en Full_Name(ABR.Nodekey(Parcours2),RG)=Full_Name(ABR.Nodekey(Parcours1),RG) then
                    --
Si le nom de la racine de Parcours 2 n'est pas nul et qu'il est égal à celui de la racine d
e Parcours 1
                        B2:=True; --
B2 est vrai est signifiera que l'homonyme a été retrouvé
                    else
                        if not Est_Nul_AG(ABR.Fils_Gauche(Parcours2)) then --
Si le fils gauche de Parcours 2 n'est pas nul
                            Intermediaire1(Parcours1,ABR.Fils_Gauche(Parcours2),AG,RG); --
On recherche dans le fils gauche de Parcours 2
                        end if;
                        if not Est_Nul_AG(ABR.Fils_Droit(Parcours2)) then --
Si le fils droit de Parcours 2 n'est pas nul
                            Intermediaire1(Parcours1,ABR.Fils_Droit(Parcours2),AG,RG); --
On recherche dans le fils droit de Parcours 2
                        end if;
                    end if;
                end if;
            end Intermediaire1;
        procedure Intermediaire2(Parcours1: in ABR.T_Branch; AG: in ABR.T_Branch; RG: in T_
Access) is
            begin
                if not Est_Nul_AG(AG) then
                    Intermediaire1(Parcours1,Node2,AG,RG); --
Voir si le nom de la racine de Parcours 1 figure quelque part dans tout le sous-arbre de m2
                    if B2 then --Si B2 est devenu True alors l'homonyme a été trouvé
                        B1:=True; --B1 signifiera maintenant que l'homonyme a été trouvé
                    else
                        if (not Est_Nul_AG(ABR.Fils_Gauche(Parcours1))) then --
Si le fils gauche de Parcours 1 n'est pas nul
                            Intermediaire2(ABR.Fils_Gauche(Parcours1),AG,RG); --
On teste maintenant si le nom du fils gauche figure quelquepart dans le sous-arbre de m2
                        end if;
                        if (not Est_Nul_AG(ABR.Fils_Droit(Parcours1))) then --
Si le fils droit de Parcours 1 n'est pas nul
```

```
                        Intermediaire2(ABR.Fils_Droit(Parcours1),AG,RG); --
On teste si le nom du fils droit figure quelquepart dans le sous-arbre de m2
                    end if;
                end if;
            end if;
        end Intermediaire2;
    begin
        Intermediaire2(Node1,AG,RG);
        if B1 then return True; --
Si B1 est devenu vrai c'est qu'au moins un homonyme existe
        else return False;      --Sinon, il n'y a pas d'homonymes
        end if;
    end Homonymes;

    procedure Devtest(AG: in out ABR.T_Branch; RG: in out T_Access) is
        int: Integer;
        begin
            Start(20,AG,RG);
            int:=10 ;Add_Ancestor(int,'P',20,AG,RG);
            int:= 9 ;Add_Ancestor(int,'P',10,AG,RG);
            int:=15 ;Add_Ancestor(int,'M',10,AG,RG);
            int:=13 ;Add_Ancestor(int,'P',15,AG,RG);
            int:=12 ;Add_Ancestor(int,'P',13,AG,RG);
            int:=14 ;Add_Ancestor(int,'M',13,AG,RG);
            int:=17 ;Add_Ancestor(int,'M',15,AG,RG);
            int:=16 ;Add_Ancestor(int,'P',17,AG,RG);
            int:=19 ;Add_Ancestor(int,'M',17,AG,RG);
            int:=18 ;Add_Ancestor(int,'P',19,AG,RG);
            int:=30 ;Add_Ancestor(int,'M',20,AG,RG);
            int:=35 ;Add_Ancestor(int,'M',30,AG,RG);
            int:=32 ;Add_Ancestor(int,'P',35,AG,RG);
            int:=31 ;Add_Ancestor(int,'P',32,AG,RG);
            int:=33 ;Add_Ancestor(int,'M',32,AG,RG);
            int:=34 ;Add_Ancestor(int,'M',33,AG,RG);
            int:=37 ;Add_Ancestor(int,'M',35,AG,RG);
            int:=36 ;Add_Ancestor(int,'P',37,AG,RG);
            int:=38 ;Add_Ancestor(int,'M',37,AG,RG);
            Add_Name(19,To_Unbounded_String("Bob"),RG);
            Add_Name(36,To_Unbounded_String("Bob"),RG);
        end Devtest;

    begin
        Put_Line(" ");
        Put_Line("          Arbres Généalogiques et Registres d'Etat Civil          ");New_
Line;New_Line;
        Put("Voulez-vous créer un arbre généalogique et son registre ? [y/n] : ");
        Get(Commencer); New_Line; --traiter l'erreur où l'utilisateur saisit autre chose!!!
        if Commencer/='n' and Commencer/='N' then
```

```
            Skip_Line;
            Put("Commencez d'abord par saisir la valeur de la racine (nombre entier) : ");
--traiter l'erreur où l'utilisateur saisit autre chose!!!
            Get(Racine);New_Line;
            Start(Racine,AG,RG); New_Line;
            Put("Saisissez le nom complet de la racine : ");Skip_Line;New_Name:=To_Unbounde
d_String(Get_Line);New_Line;
            Add_Name(Racine,New_Name,RG);
            Put_Line("Saisissez sa date de naissance : ");
            Put("      -Le jour : "); Get(NewDay,2);New_Line;Skip_Line;
            while NewDay<1 or NewDay>31 loop
                Put("       Le jour saisi est invalide! Saisissez une valeur entre 1 et 31
: ");Get(NewDay);New_Line;Skip_Line;
            end loop;
            Put("      -Le mois : "); Get(NewMonth_Integer,2);New_Line;Skip_Line;
            while NewMonth_Integer<1 or NewMonth_Integer>12 loop
                Put("       Le mois saisi est invalide! Saisissez une valeur entre 1 et 12
: ");Get(NewMonth_Integer);New_Line;Skip_Line;
            end loop;
            Put("      -L'année : "); Get(NewYear);New_Line;
            case NewMonth_Integer is
                when 1=>NewMonth:=JANVIER;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 2=>NewMonth:=FEVRIER;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 3=>NewMonth:=MARS;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 4=>NewMonth:=AVRIL;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 5=>NewMonth:=MAI;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 6=>NewMonth:=JUIN;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 7=>NewMonth:=JUILLET;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 8=>NewMonth:=AOUT;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 9=>NewMonth:=SEPTEMBRE;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 10=>NewMonth:=OCTOBRE;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 11=>NewMonth:=NOVEMBRE;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when 12=>NewMonth:=DECEMBRE;Add_BirthD(Racine,NewDay,NewMonth,NewYear,RG);
                when others=>
                    Put_Line("Le mois saisi est invalide!");
            end case;
            Put("Saisissez son lieu de naissance : ");Skip_Line;NewBirthP:=To_Unbounded_Str
ing(Get_Line); New_Line; New_Line;
            Add_BirthP(Racine,NewBirthP,RG);
            Put_Line("Vous avez bien construit la racine de l'arbre généalogique.");New_Lin
e;New_Line;
            loop
                Put_Line("         Menu Principal           ");New_Line;
                Put_Line("0. DEV TEST");
                Put_Line("1. Vérifications");
                Put_Line("2. Opérations");
                Put_Line("3. Affichages");
                Put_Line("4. Quitter ?");New_Line;
```

```
                Put("Quelle option choisissez vous? : ");
                Get(Option);New_Line;
                New_Line;
                case Option is
                    when 0 =>
                        Devtest(AG,RG); Racine:=20;
                        Put_Line("Arbre de DEVTEST construit.");New_Line;New_Line;
                    when 1 =>
                        loop
                            Put_Line("         Menu des Vérifications          ");New_Line;
                            Put_Line("1. Vérifier si l'arbre et le registre sont vides");
                            Put_Line("2. Vérifier si une clé existe dans l'arbre et son reg
istre");
                            Put_Line("3. Vérifier si deux individus ont un ou plusieurs anc
êtres homonymes");
                            Put_Line("4. Revenir au Menu Principal");New_Line;
                            Put("Quelle option choisissez vous? : ");
                            Get(Option1); New_Line;New_Line;
                            case Option1 is
                                when 1 =>
                                    if Sont_Vides(AG,RG) then Put_Line("L'arbre et son regi
stre sont vides."); else Null; end if;
                                when 2 =>
                                    Put("Saisissez la clé que vous cherchez : "); Get(New_K
ey);New_Line;
                                    if Existe(New_Key,AG,RG) then Put_Line("La clé existe d
ans l'arbre et le registre."); else Null; end if; New_Line;New_Line;
                                when 3 =>
                                    Put("Saisissez la clé du premier individu : "); Get(New
_Key);New_Line;
                                    Put("Saisissez la clé du deuxième individu : ");Get(New
er_Key);New_Line;
                                    if not Existe(New_Key,AG,RG) or not Existe(Newer_Key,AG
,RG) then
                                        Put_Line("Clé(s) introuvable(s) !"); New_Line; New_
Line;
                                    else
                                        if Homonymes(New_Key,Newer_Key,AG,RG) then
                                            Put_Line("Ces individus ont bien un ou plusieur
s ancêtres homonymes."); New_Line; New_Line;
                                        else
                                            Put_Line("Ces individus n'ont pas d'ancêtres ho
monymes."); New_Line; New_Line;
                                        end if;
                                    end if;
                                when 4 =>
                                    MenuPrecedent:=True;
```

```
                                    Put_Line("Retour au menu précédent..."); New_Line;New_L
ine;
                                when others =>
                                    Put_Line("Saisissez une option valide."); New_Line;
                            end case;
                            New_Line;
                        exit when MenuPrecedent;
                        end loop;
                        MenuPrecedent:=False;
                    when 2 =>
                        loop
                            Put_Line("         Menu des Opérations           ");New_Line;
                            Put_Line("0. Multiplier toutes les clés par 10");
                            Put_Line("1. Ajouter un ancêtre");
                            Put_Line("2. Supprimer un ancêtre");
                            Put_Line("3. Modifier la clé d'un ancêtre");
                            Put_Line("4. Modifier le sexe d'un ancêtre");
                            Put_Line("5. Ajouter/Modifier toutes les informations d'un ancê
tre");
                            Put_Line("6. Ajouter/Modifier le nom complet d'un ancêtre");
                            Put_Line("7. Ajouter/Modifier la date de naissance d'un ancêtre
");
                            Put_Line("8. Ajouter/Modifier le lieu de naissance d'un ancêtre
");
                            Put_Line("9. Revenir au Menu Principal");New_Line;
                            Put("Quelle option choisissez vous? : ");
                            Get(Option2); New_Line; New_Line;
                            case Option2 is
                                when 0=>
                                    ABR.Multiplier_10(AG);
                                    RG_Multiplier_10(RG);
                                    Racine:=Racine*10;
                                    Put_Line("Toutes les clés ont été multipliées par 10.")
;New_Line;New_Line;
                                when 1 =>
                                    Put("Saisissez la clé du nouvel ancêtre : "); Get(New_K
ey);New_Line;
                                    Put("Saisissez la clé de son descendant : "); Get(New_K
ey_Desc);New_Line;
                                    Put("Saisissez le lien de parenté [M/P]: "); Get(New_Do
nnee);New_Line;
                                    NbrFils_Avant:=Nombre_Ancetres(New_Key_Desc,AG);
                                    Affecter_Rech_Noeud_AG(New_Key_Desc,AG,NoeudAncetre);
                                    Add_Ancestor(New_Key,New_Donnee,New_Key_Desc,AG,RG);
                                    Put("Vérification que l'ajout a été fait... ");
                                    if ABR.Nodekey(AG)=10*Racine then
                                        Racine:=ABR.Nodekey(AG);
                                        if New_Key>New_Key_Desc then
```

```
                                        New_Key:=(New_Key_Desc*10)+5;
                                    else
                                        New_Key:=(New_Key_Desc*10)-5;
                                    end if;
                                end if;
                                NbrFils_Apres:=Nombre_Ancetres(ABR.Nodekey(NoeudAncetre
),AG);
                                if NbrFils_Avant=NbrFils_Apres-1 then
                                    Put("Ajout réussi.");New_Line;
                                    Put("Voulez-
vous attribuer à la clé des informations ? [y/n] : "); Get(AddInfo);New_Line;
                                    if AddInfo/='n' and AddInfo/='N' then
                                        Put("Saisissez le nom complet du nouvel ancêtre
 : ");Skip_Line;New_Name:=To_Unbounded_String(Get_Line);New_Line;
                                        Add_Name(New_Key,New_Name,RG);
                                        Put_Line("Saisissez sa date de naissance : ");
                                        Put("      -
Le jour : "); Get(NewDay);New_Line;Skip_Line;
                                        while NewDay<1 or NewDay>31 loop
                                            Put("       Le jour saisi est invalide! Sai
sissez une valeur entre 1 et 31 : ");Get(NewDay);New_Line;Skip_Line;
                                        end loop;
                                        Put("      -
Le mois : "); Get(NewMonth_Integer);New_Line;Skip_Line;
                                        while NewMonth_Integer<1 or NewMonth_Integer>12
 loop
                                            Put("       Le mois saisi est invalide! Sai
sissez une valeur entre 1 et 12 : ");Get(NewMonth_Integer);New_Line;Skip_Line;
                                        end loop;
                                        Put("      -
L'année : "); Get(NewYear);New_Line;
                                        while (Birth_Year(New_Key_Desc,RG) - NewYear)<1
6 loop
                                            Put_Line("Un parent doit être plus âgé que
son descendant d'au moins 16 ans!");
                                            Put("Saisissez une année positive et inféri
eure ou égale à" & Integer'Image(Birth_Year(New_Key_Desc,RG)-
16) & " : "); Get(NewYear);New_Line;
                                        end loop;
                                        case NewMonth_Integer is
                                            when 1=>NewMonth:=JANVIER;Add_BirthD(New_Ke
y,NewDay,NewMonth,NewYear,RG);
                                            when 2=>NewMonth:=FEVRIER;Add_BirthD(New_Ke
y,NewDay,NewMonth,NewYear,RG);
                                            when 3=>NewMonth:=MARS;Add_BirthD(New_Key,N
ewDay,NewMonth,NewYear,RG);
                                            when 4=>NewMonth:=AVRIL;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);
```

```
                                            when 5=>NewMonth:=MAI;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);
                                            when 6=>NewMonth:=JUIN;Add_BirthD(New_Key,N
ewDay,NewMonth,NewYear,RG);
                                            when 7=>NewMonth:=JUILLET;Add_BirthD(New_Ke
y,NewDay,NewMonth,NewYear,RG);
                                            when 8=>NewMonth:=AOUT;Add_BirthD(New_Key,N
ewDay,NewMonth,NewYear,RG);
                                            when 9=>NewMonth:=SEPTEMBRE;Add_BirthD(New_
Key,NewDay,NewMonth,NewYear,RG);
                                            when 10=>NewMonth:=OCTOBRE;Add_BirthD(New_K
ey,NewDay,NewMonth,NewYear,RG);
                                            when 11=>NewMonth:=NOVEMBRE;Add_BirthD(New_
Key,NewDay,NewMonth,NewYear,RG);
                                            when 12=>NewMonth:=DECEMBRE;Add_BirthD(New_
Key,NewDay,NewMonth,NewYear,RG);
                                            when others=>
                                                Put_Line("Le mois saisi est invalide!")
;
                                        end case;
                                        Put("Saisissez son lieu de naissance : ");Skip_
Line;NewBirthP:=To_Unbounded_String(Get_Line); New_Line; New_Line;
                                        Add_BirthP(New_Key,NewBirthP,RG);
                                        Put_Line("Vous avez bien ajouté le nouvel ancêt
re et ses informations à l'arbre et au registre.");New_Line;New_Line;
                                    else
                                        Put_Line("Retour au menu des opérations...");Ne
w_Line;New_Line;
                                    end if;
                                else
                                    Put_Line("Ajout échoué!");New_Line;New_Line;
                                end if;
                            when 2 =>
                                Put("Saisissez la clé de l'ancêtre que vous voulez supp
rimer : "); Get(New_Key);New_Line;
                                Delete(New_Key,AG,RG);
                                if not (Est_Present(New_Key,AG) and Existe_RG(New_Key,R
G)) then
                                    Put_Line("Suppression effectuée."); New_Line;New_Li
ne;
                                else
                                    Put_Line("Suppression échouée!"); New_Line;New_Line
;
                                end if;
                            when 3 =>
                                Put("Saisissez la clé de l'ancêtre dont vous voulez cha
nger la clé : "); Get(New_Key);New_Line;
                                if Existe(New_Key,AG,RG) then
```

```
                                        Put("Saissez la nouvelle clé : ");Get(Newer_Key);Ne
w_Line;
                                        Edit_Key(New_Key,Newer_Key,AG,RG);
                                        Put_Line("Modification effectuée.");
                                        if New_Key=Racine then
                                            Racine:=ABR.Nodekey(AG);
                                        end if;
                                    end if;
                                    New_Line;New_Line;
                                when 4 =>
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez cha
nger le sexe : "); Get(New_Key);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                        Put("Saisissez le nouveau sexe [M/P]: "); Get(New_D
onnee);New_Line; --control error!
                                        Edit_Sexe(New_Key,New_Donnee,AG);
                                        Put_Line("Modification effectuée.");
                                    end if;
                                    New_Line;New_Line;
                                when 5 =>
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez att
ribuer des informations : "); Get(New_Key);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                        Put("Saisissez le nom complet de l'ancêtre : ");Ski
p_Line;New_Name:=To_Unbounded_String(Get_Line);New_Line;
                                        Add_Name(New_Key,New_Name,RG);
                                        Put_Line("Saisissez sa date de naissance : ");
                                        Put("      -
Le jour : "); Get(NewDay);New_Line;Skip_Line;
                                        while NewDay<1 or NewDay>31 loop
                                                Put("       Le jour saisi est invalide! Sai
sissez une valeur entre 1 et 31 : ");Get(NewDay);New_Line;Skip_Line;
                                            end loop;
                                            Put("      -
Le mois : "); Get(NewMonth_Integer);New_Line;Skip_Line;
                                            while NewMonth_Integer<1 or NewMonth_Integer>12
 loop
                                                Put("       Le mois saisi est invalide! Sai
sissez une valeur entre 1 et 12 : ");Get(NewMonth_Integer);New_Line;Skip_Line;
                                            end loop;
                                        Put("      -L'année : "); Get(NewYear);New_Line;
                                        while New_Key/=(ABR.Nodekey(AG)) and then (Birth_Ye
ar(ABR.Nodekey(ABR.Rech_Ancetre(New_Key,AG)),RG) - NewYear)<16 loop
                                            Put_Line("Un parent doit être plus âgé que son
descendant d'au moins 16 ans!");
                                            Put("Saisissez une année positive et inférieure
 ou égale à" & Integer'Image(Birth_Year(ABR.Nodekey(ABR.Rech_Ancetre(New_Key,AG)),RG)-
16) & " : "); Get(NewYear);New_Line;
```

```
                                    end loop;
                                    case NewMonth_Integer is
                                        when 1=>NewMonth:=JANVIER;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);
                                        when 2=>NewMonth:=FEVRIER;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);
                                        when 3=>NewMonth:=MARS;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);
                                        when 4=>NewMonth:=AVRIL;Add_BirthD(New_Key,NewD
ay,NewMonth,NewYear,RG);
                                        when 5=>NewMonth:=MAI;Add_BirthD(New_Key,NewDay
,NewMonth,NewYear,RG);
                                        when 6=>NewMonth:=JUIN;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);
                                        when 7=>NewMonth:=JUILLET;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);
                                        when 8=>NewMonth:=AOUT;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);
                                        when 9=>NewMonth:=SEPTEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);
                                        when 10=>NewMonth:=OCTOBRE;Add_BirthD(New_Key,N
ewDay,NewMonth,NewYear,RG);
                                        when 11=>NewMonth:=NOVEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);
                                        when 12=>NewMonth:=DECEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);
                                        when others=>
                                            Put_Line("Le mois saisi est invalide!");
                                    end case;
                                    Put("Saisissez son lieu de naissance : ");Skip_Line
;NewBirthP:=To_Unbounded_String(Get_Line); New_Line; New_Line;
                                    Add_BirthP(New_Key,NewBirthP,RG);
                                    Put_Line("Vous avez bien ajouté/modifié toutes les
informations de l'ancêtre.");
                                end if;
                                New_Line;New_Line;
                            when 6 =>
                                Put("Saisissez la clé de l'ancêtre auquel vous voulez a
ttribuer un nom : "); Get(New_Key);New_Line;
                                if Existe(New_Key,AG,RG) then
                                    Put("Saisissez le nom complet : ");Skip_Line; New_N
ame:=To_Unbounded_String(Get_Line);New_Line; --control error!
                                    Add_Name(New_Key,New_Name,RG);
                                    Put_Line("Ajout/Modification effectuée.");
                                end if;
                                New_Line;New_Line;
                            when 7 =>
```

```
                                    Put("Saisissez la clé de l'ancêtre auquel vous voulez a
ttribuer une date de naissance : ");Get(New_Key);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                        Put_Line("Saisissez sa date de naissance : "); Skip
_Line;
                                        Put("      -
Le jour : "); Get(NewDay);New_Line;Skip_Line;
                                        while NewDay<1 or NewDay>31 loop
                                                Put("       Le jour saisi est invalide! Sai
sissez une valeur entre 1 et 31 : ");Get(NewDay);New_Line;Skip_Line;
                                            end loop;
                                            Put("      -
Le mois : "); Get(NewMonth_Integer);New_Line;Skip_Line;
                                            while NewMonth_Integer<1 or NewMonth_Integer>12
 loop
                                                Put("       Le mois saisi est invalide! Sai
sissez une valeur entre 1 et 12 : ");Get(NewMonth_Integer);New_Line;Skip_Line;
                                            end loop;
                                        Put("      -L'année : "); Get(NewYear);New_Line;
                                        while New_Key/=ABR.Nodekey(AG) and then (Birth_Year
(ABR.Nodekey(ABR.Rech_Ancetre(New_Key,AG)),RG) - NewYear)<16 loop
                                            Put_Line("Un parent doit être plus âgé que
son descendant d'au moins 16 ans!");
                                            Put("Saisissez une année positive et inféri
eure ou égale à" & Integer'Image(Birth_Year(ABR.Nodekey(ABR.Rech_Ancetre(New_Key,AG)),RG)-
16) & " : "); Get(NewYear);New_Line;
                                        end loop;
                                        case NewMonth_Integer is
                                            when 1=>NewMonth:=JANVIER;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 2=>NewMonth:=FEVRIER;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 3=>NewMonth:=MARS;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 4=>NewMonth:=AVRIL;Add_BirthD(New_Key,NewD
ay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 5=>NewMonth:=MAI;Add_BirthD(New_Key,NewDay
,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 6=>NewMonth:=JUIN;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 7=>NewMonth:=JUILLET;Add_BirthD(New_Key,Ne
wDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 8=>NewMonth:=AOUT;Add_BirthD(New_Key,NewDa
y,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 9=>NewMonth:=SEPTEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                            when 10=>NewMonth:=OCTOBRE;Add_BirthD(New_Key,N
ewDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
```

```
                                        when 11=>NewMonth:=NOVEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                        when 12=>NewMonth:=DECEMBRE;Add_BirthD(New_Key,
NewDay,NewMonth,NewYear,RG);Put_Line("Ajout/Modification effectuée.");
                                        when others=>
                                            Put_Line("Le mois saisi est invalide!");
                                    end case;
                                end if;
                                New_Line;New_Line;
                            when 8 =>
                                Put("Saisissez la clé de l'ancêtre auquel vous voulez a
ttribuer un lieu de naissance : ");Get(New_Key);New_Line;
                                if Existe(New_Key,AG,RG) then
                                    Put("Saisissez son lieu de naissance : ");Skip_Line
; NewBirthP:=To_Unbounded_String(Get_Line);New_Line;
                                    Add_BirthP(New_Key,NewBirthP,RG);
                                    Put_Line("Ajout/Modification effectuée.");
                                end if;
                                New_Line;New_Line;
                            when 9 =>
                                MenuPrecedent:=True;
                                Put_Line("Retour au menu précédent..."); New_Line;New_L
ine;
                            when others =>
                                Put_Line("Saisissez une option valide."); New_Line;
                        end case;
                    exit when MenuPrecedent;
                    end loop;
                    MenuPrecedent:=False;
                when 3 =>
                    loop
                        Put_Line("         Menu des Affichages           ");New_Line;
                        Put_Line(" 1. Afficher toutes les informations d'une clé");
                        Put_Line(" 2. Afficher le nom complet d'une clé");
                        Put_Line(" 3. Afficher la date de naissance d'une clé");
                        Put_Line(" 4. Afficher le lieu de naissance d'une clé");
                        Put_Line(" 5. Afficher les clés et noms des parents d'une clé")
;
                        Put_Line(" 6. Afficher le nombre d'ancêtres d'une clé");
                        Put_Line(" 7. Afficher les clés d'une certaine génération par r
apport à une clé");
                        Put_Line(" 8. Afficher les clés d'une certaine génération ou mo
ins par rapport à une clé");
                        Put_Line(" 9. Afficher les individus n'ayant aucun parent connu
");
                        Put_Line("10. Afficher les individus ayant un seul parent connu
");
```

```
                            Put_Line("11. Afficher les individus dont les deux parents sont
 connus");
                            Put_Line("12. Afficher l'arbre complet");
                            Put_Line("13. Afficher l'arbre à partir d'une clé");
                            Put_Line("14. Revenir au Menu Principal");New_Line;
                            Put("Quelle option choisissez vous? : ");
                            Get(Option3); New_Line;New_Line;
                            case Option3 is
                                when 1=>
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher toutes les informations : "); Get(New_Key);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                        Put("Le nom complet de la clé " & Integer'Image(New
_Key) & " est       : ");Put(Full_Name(New_Key,RG));Put(".");New_Line;
                                        Put("La date de naissance de la clé " & Integer'Ima
ge(New_Key) & " est : ");
                                        Afficher_Date(Birth_Date(New_Key,RG));
                                        Put(".");
                                        New_Line;
                                        Put("Le lieu de naissance de la clé " & Integer'Ima
ge(New_Key) & " est : "); Put(Birth_Place(New_Key,RG));Put(".");New_Line;
                                        if Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key
,AG))) and Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG))) then
                                            Put_Line("Cette clé n'a pas de parent connu!");
                                        elsif not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(
New_Key,AG))) then
                                            if ABR.NodeValue(ABR.Fils_Gauche(Rech_Noeud_AG(
New_Key,AG)))='M' then
                                                Put("Sa mère a pour clé " & Integer'Image(A
BR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Gauche(
Rech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            elsif ABR.NodeValue(ABR.Fils_Gauche(Rech_Noeud_
AG(New_Key,AG)))='P' then
                                                Put("Son père a pour clé " & Integer'Image(
ABR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Gauche(
Rech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            end if;
                                        end if;
                                        if not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(New_
Key,AG))) then
                                             if ABR.NodeValue(ABR.Fils_Droit(Rech_Noeud_AG(
New_Key,AG)))='M' then
                                                Put("Sa mère a pour clé " & Integer'Image(A
BR.Nodekey(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Droit(R
ech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
```

```
                                        elsif ABR.NodeValue(ABR.Fils_Droit(Rech_Noeud_A
G(New_Key,AG)))='P' then
                                            Put("Son père a pour clé " & Integer'Image(
ABR.Nodekey(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                             Put(Full_Name(ABR.Nodekey(ABR.Fils_Droit(R
ech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                        end if;
                                    end if;
                                end if;
                                New_Line;Put_Line("Retour au menu des affichages dans 3
 secondes.."); delay 3.0;
                                New_Line;New_Line;
                            when 2 =>
                                 Put("Saisissez la clé de l'ancêtre dont vous voulez af
ficher le nom complet : "); Get(New_Key);New_Line;
                                 if Existe(New_Key,AG,RG) then
                                     Put("Le nom complet de la clé " & Integer'Image(Ne
w_Key) & " est : ");Put(Full_Name(New_Key,RG));Put(".");New_Line;
                                 end if;
                                 New_Line;Put_Line("Retour au menu des affichages dans
3 secondes.."); delay 3.0;
                                 New_Line;New_Line;
                            when 3 =>
                                Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher la date de naissance : "); Get(New_Key);New_Line;
                                 if Existe(New_Key,AG,RG) then
                                     Put("La date de naissance de la clé " & Integer'Im
age(New_Key) & " est : ");
                                     Afficher_Date(Birth_Date(New_Key,RG));
                                     Put(".");
                                 end if;
                                 New_Line;Put_Line("Retour au menu des affichages dans
3 secondes.."); delay 3.0;
                                 New_Line;New_Line;
                            when 4 =>
                                Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher le lieu de naissance : "); Get(New_Key);New_Line;
                                 if Existe(New_Key,AG,RG) then
                                     Put("Le lieu de naissance de la clé " & Integer'Im
age(New_Key) & " est : "); Put(Birth_Place(New_Key,RG));Put(".");
                                 end if;
                                 New_Line;Put_Line("Retour au menu des affichages dans
3 secondes.."); delay 3.0;
                                 New_Line;New_Line;
                            when 5 =>
                                Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher les informations des parents : "); Get(New_Key);New_Line;
                                if Existe(New_Key,AG,RG) then
```

```
                                        if Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key
,AG))) and Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG))) then
                                            Put_Line("Cette clé n'a pas de parent connu!");
                                        elsif not Est_Nul_AG(ABR.Fils_Gauche(Rech_Noeud_AG(
New_Key,AG))) then
                                            if ABR.NodeValue(ABR.Fils_Gauche(Rech_Noeud_AG(
New_Key,AG)))='M' then
                                                Put("Sa mère a pour clé " & Integer'Image(A
BR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Gauche(
Rech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            elsif ABR.NodeValue(ABR.Fils_Gauche(Rech_Noeud_
AG(New_Key,AG)))='P' then
                                                Put("Son père a pour clé " & Integer'Image(
ABR.Nodekey(ABR.Fils_Gauche(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Gauche(
Rech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            end if;
                                        end if;
                                        if not Est_Nul_AG(ABR.Fils_Droit(Rech_Noeud_AG(New_
Key,AG))) then
                                             if ABR.NodeValue(ABR.Fils_Droit(Rech_Noeud_AG(
New_Key,AG)))='M' then
                                                Put("Sa mère a pour clé " & Integer'Image(A
BR.Nodekey(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Droit(R
ech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            elsif ABR.NodeValue(ABR.Fils_Droit(Rech_Noeud_A
G(New_Key,AG)))='P' then
                                                Put("Son père a pour clé " & Integer'Image(
ABR.Nodekey(ABR.Fils_Droit(Rech_Noeud_AG(New_Key,AG)))) & " et s'appelle    : ");
                                                 Put(Full_Name(ABR.Nodekey(ABR.Fils_Droit(R
ech_Noeud_AG(New_Key,AG))),RG));Put(".");New_Line;
                                            end if;
                                        end if;
                                    end if;
                                    New_Line;Put_Line("Retour au menu des affichages dans 3
 secondes.."); delay 3.0;
                                    New_Line;New_Line;
                                when 6 =>
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher le nombre d'ancêtres : "); Get(New_Key);New_Line;
                                     if Existe(New_Key,AG,RG) then
                                         Put("Le nombre d'ancêtres de la clé " & Integer'Im
age(New_Key) & " est : " & Integer'Image(Ancestor_Nbr(New_Key,AG)) & ".");
                                     end if;
                                     New_Line;New_Line;
                                when 7 =>
```

```
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher les ancêtres d'une certaine génération : "); Get(New_Key);New_Line;
                                    Put("Saisissez la génération : ");Get(g);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                         Same_Gen_Keys(g,New_Key,AG);
                                         New_Line;Put_Line("Retour au menu des affichages d
ans 3 secondes.."); delay 3.0;
                                    end if;
                                    New_Line;New_Line;
                                when 8 =>
                                    Put("Saisissez la clé de l'ancêtre dont vous voulez aff
icher les ancêtres d'une certaine génération ou moins: "); Get(New_Key);New_Line;
                                    Put("Saisissez la génération : ");Get(g);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                         Same_Gen_Orless(g,New_Key,AG);
                                         New_Line;Put_Line("Retour au menu des affichages d
ans 3 secondes.."); delay 3.0;
                                    end if;
                                    New_Line;New_Line;
                                when 9 =>
                                    Ensemble_Orphelins(AG);
                                when 10 =>
                                    Ensemble_Un_Parent(AG);
                                when 11 =>
                                    Ensemble_Deux_Parents(AG);
                                when 12 =>
                                    if not (Est_Nul_AG(AG) and Est_Vide_RG(RG)) then
                                        Put_Line("Affichage de l'arbre complet :");New_Line
;
                                        Print(AG);New_Line;
                                        Put_Line("Retour au menu des affichages dans 5 seco
ndes.."); delay 5.0; New_Line;
                                    else
                                        Put_Line("L'arbre et le registre sont vides!");New_
Line;New_Line;
                                    end if;
                                when 13 =>
                                    Put("Saisissez la clé de l'ancêtre à partir duquel vous
 voulez afficher l'arbre: "); Get(New_Key);New_Line;
                                    if Existe(New_Key,AG,RG) then
                                        Print_From(New_Key,AG); New_Line;
                                        Put_Line("Retour au menu des affichages dans 5 seco
ndes.."); delay 5.0;
                                    end if;
                                    New_Line;New_Line;
                                when 14 =>
                                    MenuPrecedent:=True;
```

```
                                    Put_Line("Retour au menu précédent..."); New_Line;New_L
ine;
                                when others =>
                                    Put_Line("Saisissez une option valide."); New_Line;
                            end case;
                        exit when MenuPrecedent;
                        end loop;
                        MenuPrecedent:=False;
                    when 4 =>
                        Quitter:=True;
                    when others =>
                        Put_Line("Saisissez une option valide."); New_Line;
                end case;
            exit when Quitter;
            end loop;
        else
            Null;
        end if;
        New_Line;
        Put_Line("Vous quitterez le programme dans 2 secondes .."); delay 2.0;New_Line;
        exception
            when DATA_ERROR =>
                Skip_Line;New_Line;Put_Line("Vous avez saisi un mauvais type! Redémarrage d
e l'application dans 3 secondes..");New_Line; delay 3.0; main;
    end main;
```