with Arbre\_Genealogique;            use Arbre\_Genealogique;

procedure test\_arbre\_genealogique is

    AG:Arbre\_Genealogique.Arbre\_Binaire\_Character.T\_Branch;

    procedure Exemple\_Arbre(*AG*:in out Arbre\_Genealogique.Arbre\_Binaire\_Character.T\_Branch) is

        begin

            Init\_AG(18,AG);

            Ajouter\_Ancetre(2,'P',18,AG);

            Ajouter\_Ancetre(19,'M',18,AG);

            Ajouter\_Ancetre(1,'P',2,AG);

            Ajouter\_Ancetre(15,'M',2,AG);

            Ajouter\_Ancetre(4,'P',15,AG);

            Ajouter\_Ancetre(33,'M',19,AG);

            Ajouter\_Ancetre(25,'P',33,AG);

            Ajouter\_Ancetre(42,'M',33,AG);

            Ajouter\_Ancetre(35,'P',42,AG);

        end Exemple\_Arbre;

    procedure Tester\_Exemple\_Arbre is

        begin

            Exemple\_Arbre(AG);

            --Tester la création et bonne répartition des ancêtres dans l'AG.

            pragma assert(not Est\_Nul\_AG(AG));

            pragma assert(Est\_Present(2,AG));

            pragma assert(Est\_Present(19,AG));

            pragma assert(Est\_Present(1,AG));

            pragma assert(Est\_Present(15,AG));

            pragma assert(Est\_Present(4,AG));

            pragma assert(Est\_Present(33,AG));

            pragma assert(Est\_Present(25,AG));

            pragma assert(Est\_Present(42,AG));

            pragma assert(Est\_Present(35,AG));

            pragma assert(not Est\_Present(64,AG));

            --Tester le nombre d'ancêtres d'un noeud, ce dernier est inclu dans le calcul.

            pragma assert(Nombre\_Ancetres(2,AG)=4);

            pragma assert(Nombre\_Ancetres(19,AG)=5);

            pragma assert(Nombre\_Ancetres(18,AG)=10);

            pragma assert(Nombre\_Ancetres(33,AG)=4);

            pragma assert(Nombre\_Ancetres(15,AG)=2);

            pragma assert(Nombre\_Ancetres(4,AG)=1);

            pragma assert(Nombre\_Ancetres(25,AG)=1);

       end Tester\_Exemple\_Arbre;

   procedure Tester\_Supprimer is

        begin

            Exemple\_Arbre(AG);

            --Tester la suppression

            Supprimer\_famille(35,AG);

            pragma assert(not Est\_Present(35,AG));

            Supprimer\_famille(25,AG);

            pragma assert(not Est\_Present(25,AG));

            Supprimer\_famille(4,AG);

            pragma assert(not Est\_Present(4,AG));

            Supprimer\_famille(15,AG);

            pragma assert(not Est\_Present(15,AG));

            Supprimer\_famille(1,AG);

            pragma assert(not Est\_Present(1,AG));

            Supprimer\_famille(19,AG);

            pragma assert(not Est\_Present(19,AG));

            pragma assert(not Est\_Present(33,AG));

            pragma assert(not Est\_Present(42,AG));

            Supprimer\_famille(18,AG);

            pragma assert(Est\_Nul\_AG(AG));

        end Tester\_Supprimer;

    begin

        Tester\_Exemple\_Arbre;

        Tester\_Supprimer;

    end test\_arbre\_genealogique;