with Ada.Strings.Unbounded;         use Ada.Strings.Unbounded;

with arbre\_genealogique;

package Registre is

    TYPE T\_Registre is LIMITED PRIVATE;

    TYPE T\_Access is LIMITED PRIVATE;

    TYPE T\_Mois is (JANVIER, FEVRIER, MARS, AVRIL, MAI, JUIN, JUILLET,AOUT, SEPTEMBRE, OCTOBRE, NOVEMBRE, DECEMBRE);

    TYPE T\_Date is private;

        --FONCTIONS/PROCEDURES SECONDAIRES--

        --Créer/modifier une date.

    procedure CreateDate(*D*: in out T\_Date; *Jour*: in Integer; *Mois*: in T\_Mois; *Annee*:in Integer);

        --Afficher unde date.

    procedure Afficher\_Date (*Date* : in T\_Date);

                    --TESTS--

        --Initialiser un Registre vide

    procedure Init\_RG(*RG*: in out T\_Access);

        --Initialiser un Registre.

    procedure Start\_RG(*Cle*: in Integer; *Reg*: in out T\_Access);

        --Vérifier si un registre est vide.

    function Est\_Vide\_RG(*Reg*: in T\_Access) return Boolean;

        --Vérifier si une clé existe dans le registre.

    function Existe\_RG(*Cle*: in Integer;*Reg*: in T\_Access) return Boolean;

            --FONCTIONS/PROCEDURES ELEMENTAIRES--

        --Retourne le nom d'un clé.

    function Name(*Cle*: in Integer; *Reg*:in T\_Access) return Unbounded\_String;

        --Retourne la date de naissance d'une clé.

    function BirthD(*Cle*: in Integer; *Reg*: in T\_Access) return T\_Date;

        --Retourne l'année de naissance d'une clé.

    function BirthY(*Cle*: in Integer; *Reg*: in T\_Access) return Integer;

        --Retourne le lieu de naissance d'une clé.

    function BirthP(*Cle*: in Integer; *Reg*: in T\_Access) return Unbounded\_String;

        --Multiplie toutes les clés du registre par 10.

    procedure RG\_Multiplier\_10(*RG*: in out T\_Access);

            --FONCTIONS/PROCEDURES DE RECHERCHE--

        --Rechercher le registre d'une clé.

    function Rech\_Reg(*Cle*: in Integer;*Reg*: in T\_Access) return T\_Access;

                --AJOUT/SUPPRESSION--

        --Ajouter une clé au registre. L'ajout sera dans l'ordre croissant pour faciliter l'accès.

    procedure AddKey(*Cle*: in Integer; *Reg*: in out T\_Access);

        --Supprimer une clé.

    procedure Delete\_RG(*Cle*:in Integer; *Reg*: in out T\_Access);

        --Détruire le registre.

    procedure Detruire\_RG(*RG*: in out T\_Access);

                --MODIFICATIONS--

        --Modifier la valeur d'une clé.

    procedure ModifyKey(*Cle*: in Integer; *NewCle*: in Integer; *Reg*: in out T\_Access);

        --Attribuer/Modfier un nom COMPLET à une clé.

    procedure AddName(*Cle*: in Integer; *Nom*: in Unbounded\_String; *Reg*:in out T\_Access);

        --Attribuer/Modifier une date de naissance à une clé. L'age sera attribué automatiquement.

    procedure AddBirthD(*Cle*: in Integer; *Jour*: in Integer; *Mois*: in T\_Mois; *Annee*:in Integer; *Reg*: in out T\_Access);

        --Attribuer/Modifier un lieu de naissance à une clé.

    procedure AddBirthP(*Cle*: in Integer; *Lieu*: in Unbounded\_String; *Reg*: in out T\_Access);

        --Ajouter un conjoint à une clé.

    procedure Ajouter\_Conjoint(*Cle,Conjoint*: in Integer; *RG*: in out T\_Access); --####SECONDE PARTIE####

PRIVATE

    TYPE T\_Date is

        RECORD

            Jour : Integer;

            Mois : T\_Mois;

            Annee : Integer;

        end RECORD;

    TYPE T\_Access is ACCESS T\_Registre;

    TYPE T\_Registre is

        RECORD

            Cle: Integer;

            Nom\_Complet: Unbounded\_String;

            Date\_Naissance: T\_Date;

            Lieu\_Naissance: Unbounded\_String;

            Age: Integer;

            Conjoints: arbre\_genealogique.Arbre\_Binaire\_Character.Piles\_Cle.T\_Pile;

            Suivant: T\_Access;

        end RECORD;

end Registre;