AP-12
Programmation Ada - TP 4 - Paquetages

L'objectif de ce TP est de :

- *utiliser des paquetages contenant les déclarations des types et des fonctions et procédures nécessaires*
- *poursuivre les apprentissages du TP précédent et écrire quelques algorithmes de calcul sur le temps*

Contexte : calculs autour du temps (suite)…

- Créez un nouveau répertoire **tp04** dans votre répertoire **ada**
- Copiez dans ce répertoire le fichier **p_dates.ads** disponible dans ***/users/info/pub/1a/ada/tp04***
- Insérez (si vous ne l'avez pas déjà fait) la commande de compilation **cada** dans votre fichier d'alias (.bashrc *ou* .bash_aliases)
  alias cada = "gnatgcc -c -gnatv -gnato" (cette commande permet de compiler une unité de programme séparément des autres.)

1. Première étape : paquetage p_dates

*Dans ce TP vous développez et utilisez un **paquetage** (unité de compilation séparée) appelé* **p_dates**.

- *Un paquetage est constitué de deux fichiers : un fichier de **spécifications** (nomdupaquetage**.ads**) et un fichier constituant le **corps** du paquetage (nomdupaquetage**.adb**)*
- *Les deux fichiers constitutifs d'un paquetage peuvent être compilés séparément avec la commande* **cada**.

> ***RAPPEL :***
> - ✓ *Si une fonction ou procédure est déclarée dans la partie spécifications, son corps (code) doit être développé dans la partie corps du paquetage pour qu'il n'y ait pas d'erreur de compilation.*
> - ✓ *les types, sous-types, constantes, variables globales, etc. déclarés dans le fichier de spécifications d'un paquetage ne doivent pas être déclarés dans le fichier qui constitue le corps de ce même paquetage !*

1.1. Etude du contenu de votre répertoire

Le fichier **p_dates.ads** correspond à la partie **spécification** du paquetage **p_dates**. Il contient les déclarations de types et sous-types que vous aviez effectuées pour le TP n°3 ainsi que des premières fonctions et procédures de ce TP. Vous le compléterez au fur et à mesure des questions en y écrivant les en-têtes de fonctions ou procédures à développer.

```
with p_esiut; use p_esiut;
package p_dates is

 -- 1 : sous-types-------------------------------------------------------------
 subtype T_Jour is Positive range 1..31;
 subtype T_Mois is Positive range 1..12;

-- 2 : types structurés-------------------------------------------------------
 type Tr_Date is record
    Jour : T_Jour;
    Mois : T_Mois;
    An : Positive;
 end record;

--3 : types énumérés-------------------------------------------------------
 type T_JourSemaine is (lundi,mardi,mercredi,jeudi,vendredi,samedi,dimanche);

 type T_NomMois is (janvier,fevrier,mars,avril,mai,juin,juillet,aout,septembre,
 octobre,novembre,decembre);

 -- 4 : instanciation de p_enum pour l'utilisation des types énumérés
 package P_Joursemaine_IO is new P_Enum(T_JourSemaine); use P_Joursemaine_IO;

 package P_Mois_IO is new P_Enum(T_NomMois); use P_Mois_IO;
-------------------------------------------------------------------------------
 --5 : fonctions et procédures
 -------------------------------------------------------------------------------
 procedure Affichedate(Date : in TR_Date) ;
 --{} => {Date est affichée à l'écran sous la forme : 12 janvier 1989}
function Siecle(Date : in TR_Date) return Boolean;
--{} => {vrai si l'année de Date est un début de siècle (ex : 900, 1500, 2000)}
end p_dates;
```

1.2. Fichier p_dates.adb :

Ce fichier doit correspondre à la partie **corps** du paquetage **p_dates**.

a) Commencez par le créer :
- Faites une copie du fichier **p_dates.ads**, renommez-la en **p_dates.adb**
- Insérez le mot-clé **body** entre le mot-clé **package** et le nom du paquetage
- Supprimez les clauses d'utilisation du paquetage **p_esiut,** du paquetage **p_enum** et les déclarations de sous-type et de types (car les déclarations faites dans le **.ads** d'un paquetage sont visibles dans son **.adb**).

b) Ecrivez le corps de la procédure **AfficheDate** et de la fonction **Siecle**

c) Compilez avec **cada**

2. Deuxième étape : utilisation du paquetage p_dates dans un programme

2.1. Programme principal : calcultemps.adb

Créez un nouveau fichier* calcultemps.adb *et dans ce fichier :

a) Ecrivez les clauses d'utilisation des paquetages **p_esiut** et **p_dates**

b) Ecrivez des instructions permettant de tester les procédures et fonctions du paquetage **p_dates** :
- → *Déclaration d'une variable de type* **TR_Date** *et de nom* **Date1**
- → *Saisie de la date* **Date1** *par l'utilisateur*
- → *Affichage de cette date*
- → *Ecriture d'un message disant si cette date est ou non un siècle*

c) Compilez le programme principal (avec **gnatmake** ou son alias **gm**) et exécutez-le (*testez-le plusieurs fois avec des dates différentes*)

d) Corrigez vos erreurs si nécessaire…

2.2. Autres fonctions de calcul sur les dates

a) Dans **p_dates.ads**, écrivez les en-têtes des fonctions booléennes suivantes :

```
function Bissextile(Date : in TR_Date) return Boolean;
--{} => {vrai si l'année de Date est une année bissextile (voir indications)}

function NbjoursAn(Date : in TR_Date) return Positive;
--{} => {résultat = nombre de jours de l'année de Date}
```

- *Enregistrez et compilez à nouveau cette partie du paquetage*
- *Dans* **p_dates.adb,** *écrivez le corps des deux fonctions*

INDICATIONS :

→ Un siècle débute par une année multiple de 100 (ex : l'an 200 a débuté le 3ème siècle, l'an 1500 a débuté le 16ème siècle, l'an 2000, le 21ème siècle, etc.)

→ Une année bissextile comporte 366 jours (29 jours en février) contre 365 jours pour les autres années

→ Pour être bissextile, une année est soit :
- Une année qui ne débute pas un siècle mais qui est multiple de 4 (ex : 2012)
- Un début de siècle et dans ce cas, le quotient dans la division par 100 doit être un multiple de 4 (ex : 2000 était bissextile (20 est divisible par 4) alors que 1900 ne l'était pas (19 n'est pas divisible par 4)

→ Toute autre année n'est pas bissextile…

- *Enregistrez et compilez cette partie du paquetage.*

b) Dans le programme principal **calcultemps.**adb
- *Ecrivez des instructions permettant de tester les deux nouvelles fonctions du paquetage* **p_dates** :
  - → *Message indiquant si la date* **Date1** *est bissextile*
  - → *Affichage du nombre de jours de l'année de* **Date1**
- *Compilez le programme (avec la commande* **gnatmake** *ou son alias* **gm***), exécutez-le et testez-le en entrant successivement les années 2000 (bissextile), 2003 (non bissextile), 2010 (non bissextile), 1900 (non bissextile) et 2012 (bissextile).*

3. Autres fonctions et/ou procédures

3.1. Paquetage p_dates

a) *Fonction entière Nbjours_Mois*

- *Ecrivez l'en-tête de la fonction* **Nbjours_Mois**, *dans* **p_dates.ads**

```
function Nbjours_Mois(Date: in TR_Date) return T_jour;
--{} => {nombre de jours du mois de Date}
```

- *Compilez* **p_dates.ads**
- *Dans* **p_dates.adb**, *écrivez le corps de cette fonction puis compilez.*

3.2. Programme principal : calcultemps.adb

- *Ecrivez les instructions qui permettent d'afficher le nombre de jours du mois de* **Date1**
- *Compilez à nouveau votre programme et exécutez-le*

3.3. Paquetage p_dates

a) *Fonction entière* Nbjours_Depuis0101

- *Ecrivez l'en-tête de la fonction* **Nbjours_Depuis0101**, *dans* **p_dates.ads**

```
function Nbjours_Depuis0101(Date : in TR_Date) return natural;
--{} => { résultat = nombre de jours écoulés depuis le 1er janvier de l'année de Date
--         exemple : si Date = 25/01/2008, résultat = 24}
```

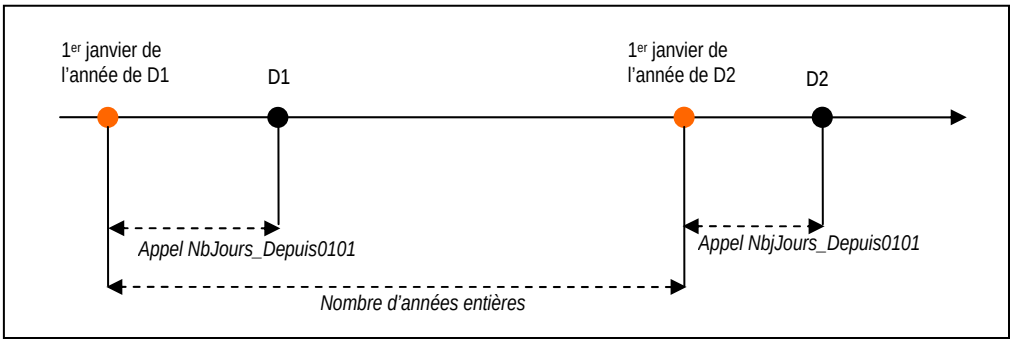
- *Dans* **p_dates.adb**, *écrivez le corps de cette fonction (testez cette fonction dans le programme principal)*

b) *Fonction entière* Delta_Jours

- *Ecrivez l'en-tête de la fonction* **Delta_Jours**, *dans* **p_dates.ads**

```
function Delta_Jours(D1, D2: in TR_Date) return natural;
--{D1 anterieure à D2} => {nombre de jours d'écart entre D1 et D2}
```

- *Dans* **p_dates.adb**, *écrivez le corps de cette fonction*
  INDICATIONS *(en considérant que D1 est antérieure à D2) : vous pouvez écrire un algorithme simple et efficace basé sur le schéma suivant :*



c) *Programme principal :* calcultemps.adb

- *Ecrivez les instructions qui permettent d'afficher le nombre de jours entre deux dates* **Date1** *et* **Date2**
  (INDICATION : pour vérifier l'ordre des deux dates pensez à utiliser la fonction **Anterieure** *vue en TD)*
- *Compilez à nouveau votre programme et exécutez-le*
  TESTS A EFFECTUER :

D1	D2	Nbjours écart
01/02/2000	25/05/2008	*3036*
13/02/2000	11/05/2008	*3010*
25/08/2001	12/01/2009	*2697*
13/09/2007	25/12/2008	*469*
05/09/2008	01/01/2009	*118*

4. … et la surprise…

4.1. Quel jour de la semaine fêterez-vous votre prochain anniversaire ?

Sachant que **le 1er septembre 2012 était un samedi**, écrivez dans **un nouveau programme** les instructions qui permettront de découvrir quel sera le jour de la semaine de votre prochain anniversaire.