

# Date du lendemain [dt08] - Exercice résolu

Karine Zampieri, Stéphane Rivière

Unisciel  algoprogram  Version 21 mai 2018

## Table des matières

<b>1</b>	<b>Énoncé</b>	<b>2</b>
<b>2</b>	<b>Algorithmique, Programmation</b>	<b>2</b>
2.1	Analyse . . . . .	2
2.2	Fonction bissextile (test de bissextilité) . . . . .	2
2.3	Fonction dernierJour (dernier jour d'un mois et année) . . . . .	3
2.4	Test : Dernier jour d'un mois et année donnés . . . . .	3
2.5	Type DateJMA . . . . .	4
2.6	Procédure calculerDemainDT (lendemain d'une date) . . . . .	4
2.7	Fonction saisirGregDT (saisie d'une date) . . . . .	4
2.8	Procédure afficherDT (affichage d'une date) . . . . .	4
2.9	Test : Lendemain d'une date . . . . .	4
2.10	Application : Nombre de jours écoulés . . . . .	5

## Python - Date du lendemain (TP)



**Mots-Clés** Dates et Heures ■

**Requis** Axiomatique impérative (sauf Fichiers) ■

**Difficulté** ●○○



### Objectif

Cet exercice calcule le lendemain d'une date (jour, mois, année) puis calcule le nombre de jours écoulés entre deux dates.

# 1 Énoncé

Une date est mémorisée dans trois variables de type entier, une pour le numéro du jour, une pour le numéro de mois et une pour le millésime de l'année. Par exemple, la date du 12 février 2013 est composée des trois entiers (12, 2, 2013).

## Objectif

Demander les trois entiers composant la date d'un jour puis calculer et afficher les trois entiers composant la date du lendemain. En application, calculer le nombre de jours écoulés entre deux dates.

# 2 Algorithmique, Programmation

## 2.1 Analyse

Ce problème se simplifie si on essaie de dégager des règles de changement de date sans s'attacher aux cas particuliers des jours et des mois précis. En effet, il apparaît qu'il y a trois types de changement de dates :

- **année suivante** si la date saisie est le dernier jour de décembre
- **mois suivant** si la date est celle du dernier jour d'un mois (sauf décembre)
- **jour suivant** dans les autres cas

Le point le plus délicat est celui du changement de mois (sauf décembre). Pour éviter de tester chacun des onze mois dans l'année, on aimerait trouver un critère plus général. Or ce critère existe si on est capable de calculer le *nombre de jours du mois* d'une date. Il suffit alors de comparer ce nombre avec la valeur saisie dans la variable représentant le jour. Si ces deux nombres sont égaux, alors il faut changer de mois.

Pour résoudre ce problème il faut donc gérer :

- Les années bissextiles pour le mois de février.
- Le nombre de jours dans le mois.
- Le changement de mois.
- Le changement d'année.

## 2.2 Fonction bissextile (test de bissextilité)



### Définition

Soit une année postérieure à 1582 (début du calendrier grégorien). Elle est **bissextile** si et seulement si son millésime est :

- Divisible par 4 mais **non** divisible par 100.
- **Ou** divisible par 400.

**Exemples**

- 1986 : non (non divisible par 4)
- 1988 : oui (divisible par 4 et non divisible par 100)
- 1900 : non (divisible par 4 et par 100)
- 2000 : oui (divisible par 400)



Écrivez une fonction `bissextile(an)` qui teste et renvoie `Vrai` si le millésime d'une année `an` (entier supérieur à 1582) est bissextile, `Faux` sinon.

**2.3 Fonction dernierJour (dernier jour d'un mois et année)****Propriété**

Pour les années postérieures à 1582 (année du calendrier Grégorien) :

- Les mois numéros 1, 3, 5, 7, 8, 10 et 12 ont 31 jours.
- Ceux de numéros 4, 6, 9 et 11 en ont 30.
- Le mois numéro 2 a 29 jours si l'année est bissextile, 28 dans le cas contraire.



Écrivez une fonction `dernierJour(mm, an)` qui calcule et renvoie le dernier jour d'un numéro de mois `mm` (entier compris entre 1 et 12) d'une année `an` (entier supérieur à 1582), selon l'algorithme suivant :

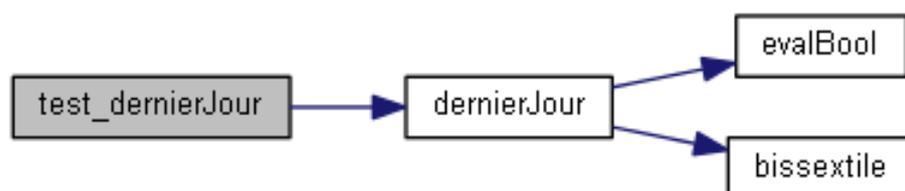
- Si le mois vaut 2 : le dernier jour est 28 (ou 29 si l'année est bissextile).
- Sinon si le mois est impair et  $\leq 7$  ou pair et  $> 7$  : le dernier jour est 31.
- Sinon le dernier jour est 30.

**2.4 Test : Dernier jour d'un mois et année donnés**

Écrivez une procédure `test_dernierJour` qui saisit un numéro de mois et le millésime d'une année puis calcule et affiche le dernier jour du mois dans l'année. Affichez les invites :

Numero de mois ([1..12])?

Millesime de l'annee (>1582)?





Testez. Exemples d'exécution :

```
Mois, annee? 2 2000  
==> Dernier jour est le 29
```

```
Mois, annee? 8 2010  
==> Dernier jour est le 31
```

## 2.5 Type DateJMA



Définissez le type `DateJMA` sous la forme d'un triplet d'entiers contenant : le numéro de jour, le numéro de mois et le millésime de l'année.

## 2.6 Procédure calculerDemainDT (lendemain d'une date)



Écrivez le **profil** d'une procédure `calculerDemainDT(dt)` qui passe au lendemain d'une `DateJMA dt` valide.



Écrivez le corps de la procédure.

## 2.7 Fonction saisirGregDT (saisie d'une date)



Écrivez une procédure `saisirGregDT(dt)` qui saisit une `DateJMA` dans `dt`. La date doit être demandée tant qu'elle n'est pas une date grégorienne valide. Affichez les invites :

```
Millesime de l'annee?  
Numero du mois?  
Numero du jour?
```

## 2.8 Procédure afficherDT (affichage d'une date)



Écrivez une procédure `afficherDT(dt)` qui affiche une `DateJMA dt` sous la forme (où `[x]` désigne le contenu de `x`) :

```
[jr]/[mm]/[an]
```

## 2.9 Test : Lendemain d'une date



Écrivez une procédure `test_demain` qui saisit une `DateJMA`, passe au lendemain puis affiche la nouvelle date.



Testez. Exemple d'exécution :

```
Millesime de l'annee : 2010  
Numero du mois : 12  
Numero du jour : 31  
==> Lendemain est le 1/1/2011
```

## 2.10 Application : Nombre de jours écoulés

Ce problème calcule et affiche le nombre de jours écoulés entre deux dates valides. Si la seconde date saisie est antérieure à la première, l'algorithme doit le signaler sans calculer le nombre de jours écoulés.



Écrivez une fonction `njoursEntreDT(d1,d2)` qui calcule et renvoie le nombre de jours écoulés entre les `DateJMA d1` et `DateJMA d2`, la première étant supposée antérieure à la deuxième.



Écrivez une fonction `posterieureDT(d1,d2)` qui teste et renvoie `Vrai` si la `DateJMA d1` est strictement postérieure à celle définie par `DateJMA d2`.



Écrivez une procédure `test_ecoules` qui demande deux `DateJMA` valides puis calcule et affiche le nombre de jours écoulés entre les deux dates.



Testez.