Corrigé du contrôle continu n°1

Gilles Peskine

Mardi 22 octobre 2002

Écrire dans le pseudo-langage décrit en cours un programme qui calcule le jour de la semaine correspondant à une date donnée dans le calendrier grégorien. On rappelle que les années bissextiles sont les années multiples de 4 *non multiples de* 100*, ainsi que les multiples de* 400*. On suppose que l'utilisateur entre une date sous forme d'un numéro de jour entre* 1 *et* 31*, d'un numéro de mois entre* 1 *et* 12 *et d'un numéro d'année positif.*

Nous noterons j le numéro du jour dans le mois, m le numéro du mois et a le numéro de l'année. L'idée de base est d'associer au jour un *numéro absolu* N, qui est le nombre de jours écoulés depuis le 31 décembre de l'an -1 : N vaut 0 pour le premier janvier de l'an 0.[1]. Ensuite nous regarderons le reste de la division de N par 7.

Pour calculer N, découpons encore le problème en deux : d'une part, le calcul du numéro absolu correspondant au premier janvier d'une année ; d'autre part, le calcul du nombre de jours écoulés entre le premier janvier et une date donnée de l'année.

Voici l'algorithme de calcul du numéro du premier janvier. On compte 365 jours pour chaque année, et on ajoute 1 pour tous les 29 février. Comme on veut le numéro du premier janvier, on ne compte pas l'éventuel 29 février de l'année en cours.

```
Fonction numéro_du_premier_janvier (valeur année : entier) Résultat entier
Variable n, précédente : entier
Début numéro_du_premier_janvier
   n ← année * 365
   précédente ← année - 1
   n ← n + (précédente div 4) - (précédente div 100) + (précédente div 400)
   Retourner n
Fin numéro_du_premier_janvier
```

Le calcul du nombre de jours écoulés entre le premier janvier et une date quelconque de l'année est un peu fastidieux (on aurait un programme un peu plus joli en utilisant un tableau). On fait le calcul pour une année non bissextile, puis on ajoute 1 s'il y a un 29 février à prendre en compte.

```
Fonction quantième_de_l'année (valeurs jour, mois, année : entiers) Résultat entier
Variable q : entier
Début quantième_de_l'année
   n ← jour - 1    // 0 pour le premier janvier
   Si mois > 1 alors n ← n + 31 Fin_si
   Si mois > 2 alors n ← n + 28 Fin_si
   Si mois > 3 alors n ← n + 31 Fin_si
   Si mois > 4 alors n ← n + 30 Fin_si
   Si mois > 5 alors n ← n + 31 Fin_si
   Si mois > 6 alors n ← n + 30 Fin_si
   Si mois > 7 alors n ← n + 31 Fin_si
   Si mois > 8 alors n ← n + 31 Fin_si
   Si mois > 9 alors n ← n + 30 Fin_si
```

[1] On fait comme si le calendrier grégorien avait duré depuis l'an 0.

```
    Si mois > 10 alors n ← n + 31 Fin_si
    Si mois > 11 alors n ← n + 30 Fin_si
    Si (année mod 4 = 0) et ((année mod 100 ≠ 0) ou (année mod 400 = 0)) alors
        Si mois >= 3 alors n ← n + 1 Fin_si    // L'année est bissextile, est-on après le 29/02 ?
    Fin_si
    Retourner n
Fin quantième_de_l'année
```

En combinant les deux algorithmes précédents, on obtient le numéro absolu N d'un jour quelconque.

```
Fonction numéro_absolu (valeurs jour, mois, années : entiers) Résultat entier
Début numéro_absolu
    Retourner numéro_du_premier_janvier(année) + quantième_de_l'année(jour, mois, année)
Fin numéro_absolu
```

Pour passer du numéro absolu au jour de la semaine, on utilise le fait que le 22 octobre 2002 est un mardi. Si A est le numéro absolu du 22 octobre 2002 et N est celui du jour entré par l'utilisateur, alors $N - A$ vaut 0 modulo 7 pour un mardi, 1 pour un mercredi, etc. La fonction quantième_de_la_semaine calcule $N - A$ modulo 7.

```
Fonction quantième_de_la_semaine (valeurs jour, mois, année : entiers) Résultat entier
Variables N, A : entiers
Début quantième_de_la_semaine
    A ← numéro_absolu(22, 10, 2002)
    N ← numéro_absolu(jour, mois, année)
    Retourner (N - A) mod 7    // N - A peut être négatif[2]
Fin quantième_de_la_semaine
```

Il n'y a plus qu'à utiliser la fonction précédente dans un programme qui lit les données du problème et affiche le résultat de façon lisible.

```
Variables jour, mois, année, s : entiers
Début programme principal
    Afficher « Entrez une date sous la forme jour, mois année. »
    Afficher « Jour : » puis lire jour
    Afficher « Mois : » puis lire mois
    Afficher « Année : » puis lire année
    s ← quantième_de_la_semaine(jour, mois, année)
    Si      s = 0 alors Afficher « C'est un mardi. »
    Sinon_si s = 1 alors Afficher « C'est un mercredi. »
    Sinon_si s = 2 alors Afficher « C'est un jeudi. »
    Sinon_si s = 3 alors Afficher « C'est un vendredi. »
    Sinon_si s = 4 alors Afficher « C'est un samedi. »
    Sinon_si s = 5 alors Afficher « C'est un dimanche. »
    Sinon_si s = 6 alors Afficher « C'est un lundi. »
    Fin_si
Fin programme principal
```

[2]En pseudo-langage ce n'est pas un problème ; en C++ le comportement des opérateurs `/` et `%` lorsque les arguments ne sont pas positifs dépend des compilateurs et des machines. Dans le cas présent, si `N - A` est négatif, on retournera `6 - ((6 - (N - A)) mod 7)`.