Faculté des Sciences et Techniques de Limoges.
Année scolaire 2004-2005.

ALGO 1.1 – Correction TD N°3.

Patrick Poulingeas.

Exercice 1.

Calcul de la date du lendemain et de la date de la veille d'un jour donné.
Variables

```
        jour,mois,année : entier
        jour_lendemain,mois_lendemain,année_lendemain : entier
        jour_veille,mois_veille,année_veille : entier
        nombre_de_jours_du_mois : entier
{

        // Saisie des données
        Afficher(« Entrer le numéro d'un jour (de 1 à 31) : »)
        Saisir(jour)
        Afficher(« Entrer le numéro d'un mois (de 1 à 12) : »)
        Saisir(mois)
        Afficher(« Entrer une année : »)
        Saisir(année)

        // Remarque : On travaille avec le calendrier grégorien

        // N.B. : on assimile les années bissextiles à celles qui sont divisibles par 4

        // Vérification que la date entrée par l'utilisateur est correcte
        si (jour < 1) ou (jour > 31)
           ou (mois < 1) ou (mois >12)
           ou ((mois = 2) et ( ( (jour > 29) et (année mod 4 = 0) )
                                ou ( (jour > 28) et (année mod 4 ≠ 0) ) ) )
           ou (année < 1583) alors
                Afficher (« Date incorrecte. »)

        sinon  // Si la date est correcte
        {

                // Calcul du nombre de jours du mois

                si (mois = 1) ou (mois = 3) ou (mois = 5)
                   ou (mois =7) ou (mois = 8) ou (mois = 10) ou (mois = 12) alors
                        nombre_de_jours_du_mois ← 31
                sinon
                        si (mois = 4) ou (mois = 6) ou (mois = 9) ou (mois = 11) alors
                                nombre_de_jours_du_mois ← 30
```

```
        sinon  // Cas de février
                si année mod 4 = 0 alors  // Cas d'une année bissextile
                        nombre_de_jours_du_mois ← 29
                sinon
                        nombre_de_jours_du_mois ← 28

// Calcul de la date du lendemain

si jour < nombre_de_jours_du_mois alors
{
        jour_lendemain ← jour + 1
        mois_lendemain ← mois
        année_lendemain ← année
}
sinon  // Cas où l'on est en fin d'un mois
{
        jour_lendemain ← 1
        si mois < 12 alors  // Cas d'un mois différent de décembre
        {
                mois_lendemain ← mois + 1
                année_lendemain ← année
        }
        sinon  // Cas où l'on est un 31 décembre
        {
                mois_lendemain ← 1
                année_lendemain ← année+1
        }
}

// Calcul de la date de la veille

si jour = 1 alors  // Si le jour est le premier du mois
        si mois = 1 alors  // Cas du 1er janvier
        {
                jour_veille ← 31
                mois_veille ← 12
                année_veille ← année – 1
        }
        sinon
                si mois = 3 alors  // Cas du 1er mars
                {
                        si année mod 4 = 0 alors  // Cas d'une année bissextile
                                jour_veille ← 29
                        sinon
                                jour_veille ← 28

                        mois_veille ← 2
                        année_veille ← année
                }
                sinon
```

```
                        {
                                si (mois = 5) ou (mois = 7)
                                   ou (mois = 10) ou (mois = 12) alors
                                        jour_veille ← 30
                                sinon
                                        jour_veille ←31

                                mois_veille ← mois -1
                                année_veille ← année
                        }
                sinon  // Cas d'un jour qui n'est pas le premier d'un mois
                {
                        jour_veille ← jour - 1
                        mois_veille ← mois
                        année_veille ← année
                }


                // Affichage des résultats
                Afficher(« Le jour du lendemain a le numéro : »,jour_lendemain)
                Afficher(« Le mois du lendemain a le numéro : »,mois_lendemain)
                Afficher(« L'année du lendemain est l'année : »,année_lendemain)

                Afficher(« Le jour de la veille a le numéro : »,jour_veille)
                Afficher(« Le mois de la veille a le numéro : »,mois_veille)
                Afficher(« L'année de la veille est l'année : »,année_veille)

        }  // correspond à l'accolade ouvrante du cas où la date est correcte
}
```

Exercice 2.

Remarques :

- On ne prend pas en compte les triangles plats isocèles et les triangles point.
- Il est évident que la condition :

  $(a + b \geq c)$ et $(b + c \geq a)$ et $(a + c \geq b)$

  est une condition nécessaire pour que le triplet de réels positifs (a,b,c) corresponde aux trois côtés d'un triangle.
  On démontre aussi que c'est une condition suffisante (par un petit raisonnement de géométrie analytique par exemple).

Détermination de la nature d'un triangle à partir de ses côtés.
Variables

```
        a,b,c : réel      // a, b et c désignent les 3 côtés du triangle
        a_carré,b_carré,c_carré : réel
{
        Afficher(« Détermination de la nature d'un triangle en fonction de ses côtés a, b et c »)

        // Saisie des données
        Afficher(« Entrer le côté a du triangle : »)
```

```
        Saisir(a)
        Afficher(« Entrer le côté b du triangle : »)
        Saisir(b)
        Afficher(« Entrer le côté c du triangle : »)
        Saisir(c)

        // Vérification du fait que les longueurs peuvent être les côtés d'un triangle
        si (a < 0) ou (b < 0) ou (c < 0) ou (a + b < c) ou (b + c < a) ou (a + c < b) alors
                Afficher(« Ces longueurs ne peuvent constituer les côtés d'un triangle. »)
        sinon  // Cas où a, b et c sont les côtés d'un triangle
               // Détermination de la nature du triangle
                si (a = b) et (b = c) et (a = c) alors  // La 3ème condition est superflue
                        Afficher(« Triangle équilatéral. »)
                sinon
                        si (a + c = b) ou (a + b = c) ou (b + c = a) alors
                                Afficher (« Triangle plat. »)
                        sinon
                        {
                                a_carré ← a**2
                                b_carré ← b**2
                                c_carré ← c**2
                                si (a_carré = b_carré + c_carré) ou (b_carré = a_carré + c_carré)
                                   ou (c_carré = a_carré + b_carré) alors
                                        si (a = b) ou (a = c) ou (b = c) alors
                                                Afficher (« Triangle rectangle isocèle. »)
                                        sinon
                                                Afficher (« Triangle rectangle. »)
                                sinon
                                        si  (a = b) ou (a = c) ou (b = c) alors
                                                Afficher (« Triangle isocèle. »)
                                        sinon
                                                Afficher(« Triangle quelconque. »)
                        }

}
```