TD3 : boucles for

ECE3 Lycée Carnot

13 octobre 2009

Boucles for

Ce TD va vous rappeler de bons souvenirs de cours de maths récents, puisque le principe d'une boucle FOR en Pascal est assez proche de celui du symbole $\sum$ en maths, à savoir « supprimer des petits points ». Une boucle FOR est donc une instruction répétitive permettant d'effectuer plusieurs fois de suite des calculs similaires. Des exemples classiques en maths sont les calculs de sommes, mais aussi les calculs de termes d'une suite définie par une formule de récurrence. L'instruction FOR obéit à la syntaxe suivante :

FOR i :=1 to n DO instruction ;

Comme en mathématiques quand on manipule une somme, la variable i est muette et vous pouvez donc lui donner n'importe quel autre nom. Elle devra bien sûr, comme toute variable, être déclarée en début de programme, avec un type integer. Toujours comme en maths, votre variable prendra toutes les valeurs entières entre la valeur initiale et la valeur finale stipulées (la valeur initiale a le droit d'être autre chose que 1 ; quant à la valeur finale, elle peut être égale à une autre variable n, par exemple choisie par l'utilisateur, comme dans mon exemple, mais aussi être égale à un entier fixe). Les instructions placées à l'intérieur de la boucle peuvent être des calculs faisant intervenir la variable i (comme en maths quand on calcule par exemple $\sum_{i=0}^{i=n} i^2$), mais il est très fortement déconseillé de modifier la valeur de i à m'intérieur de la boucle, sous peine de créer des boucles au comportement plus qu'étrange. Dernier détail, on a le droit de faire des boucles où la valeur de i diminue au lieu d'augmenter via la syntaxe :

FOR i :=n downto 1 DO instruction ;

Naturellement, il faudra dans ce cas que la valeur finale soit plus petite que la valeur initiale. Un premier exemple de programme faisant intevenir une boucle FOR :

```
PROGRAM suite ;
VAR u : real ;
        i,n : integer ;
BEGIN
WriteLn('Choisissez la valeur de n') ;
ReadLn(n) ;
u := 1 ;
FOR i := 1 to n do u := u/2 + 1/u ;
WriteLn('u_n=',u) ;
END.
```

Dans le cas où l'on souhaite effectuer plusieurs instructions à chaque étape de la boucle, on encadrera ces instructions par un BEGIN et un END (sinon, seule la première sera effectuée à chaque étape, et les suivantes seulement quand la boucle sera terminée).

Petits exercices

1. Essayer de comprendre ce que calcule exactement le programme précédent.
2. Écrire un programme calculant la somme des entiers entre 1 et n, pour un entier n choisi par l'utilisateur, sans utiliser la formule du cours (sinon c'est trop facile). Faire la même chose pour calculer $\sum_{k=1}^{k=n} \frac{1}{k}$ (là, au moins, vous n'avez pas de formule).
3. Écrire un programme qui calcule la valeur de $n!$, pour un entier n choisi par l'utilisateur.
4. Écrire un programme qui calcule et affiche les n premières valeurs de la suite (u_n) définie par $u_n = \sum_{k=0}^{k=n} \frac{1}{k!}$.