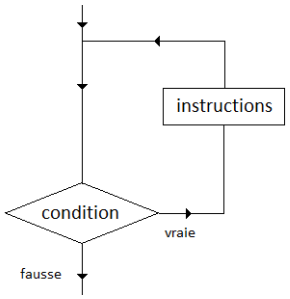
TP3

Les boucles `while...do...end`

En algorithmique, on peut être amené à répéter un bloc d'instruction tant qu'une condition est vérifiée. On utilise dans ce cas une boucle `while`. La syntaxe est la suivante :

```
while condition do
  instructions
end
```



Si la *condition* est vérifiée, alors les *instructions* sont effectuées, sinon elles sont ignorées. Le `while` et le `end` sont indispensables, le `do` est facultatif.

Attention.

Il faut veiller à ce que la boucle se termine (et donc que la *condition* soit fausse) en un nombre fini d'étapes.

Si la boucle `while` ne se termine jamais, il est possible d'interrompre ou d'abandonner l'exécution de la procédure à partir du menu de la console dans la rubrique **Contrôle**.

Exercice 1 1. Entrer dans l'éditeur les instructions suivantes :

```
n=input('Donner une valeur de n : ')
S=0
k=1
while k<=n do
    S=S+k
    k=k+1
end
disp(S,'S=')
```

Tester avec différentes valeurs pour n. A quoi correspond la valeur de S à la fin de la boucle ? Vérifier à l'aide d'une formule sur les sommes usuelles.

On dispose ainsi de deux méthodes pour calculer une somme :

- *Soit en utilisant une boucle `for` (voir le TP2).*
- *Soit en utilisant une boucle `while` (comme ci-dessus).*

2. Proposer deux procédures, une utilisant une boucle `for` et une utilisant une boucle `while`, pour calculer les sommes suivantes :

$$(i)\ \sum_{k=1}^{n}(3k^2+2k+1) \qquad (ii)\ \sum_{k=1}^{n}\frac{2^{2k-1}}{3^{k+1}} \qquad (iii)\ \sum_{k=n}^{3n}k$$

Calculer ces sommes "à la main". Vérifier que les résultats obtenus sont cohérents avec les résultats que peut proposer Scilab.

Exercice 2 Écrire une suite d'instructions qui demande à l'utilisateur un nombre inférieur ou égal à 100, et, en cas de réponse incorrecte, affiche "SVP un nombre inférieur à 100" jusqu'à ce que la réponse soit bonne.

Exercice 3 Scilab permet de générer de manière aléatoire un nombre réel compris strictement entre 0 et 1 avec l'instruction `rand()`.

1. Écrire une instruction en Scilab permettant d'obtenir un nombre entier aléatoire entre 0 et 5 (on utilisera la fonction partie entière qui est donnée par la commande `floor`).
2. Modifier l'instruction précédente pour obtenir un nombre entier aléatoire entre 1 et 6.
3. Écrire une suite d'instructions qui simule le lancé d'un dé à 6 faces et qui affiche les valeurs obtenues tant que la face 6 n'est pas sortie.