TD 1 : Algorithmique

1) Aiguiller

Ecrire l'algorithme qui permet de lire 1 réel x , un caractère qui sera ‘+’ ou ‘-‘ et un réel y, et qui effectue l’addition ou la soustraction suivant la valeur du caractère. Exemple, les données
50 – 12
doivent donner l’affichage
38
On traitera les erreurs de saisie sur l’opérateur (caractère différent du ‘+’ ou du ‘-‘)

2) Enumérer

a) Ecrire l'algorithme qui lit un entier n et affiche les nombres entiers de 1 à n
b) qui affiche les nombres de n à 1

3) Compter

Ecrire l'algorithme qui lit caractère par caractère une phrase terminée par un point, et qui compte le nombre de 'e' dans cette phrase.

4) Accumuler

Ecrire l'algorithme qui lit une suite d'entiers positifs terminée par -1 et calcule la moyenne de cette suite.

5) Plus petite valeur

Ecrire l'algorithme qui lit une suite d'entiers positifs terminée par -1 et trouve le plus petit élément de la suite.

TD 1 : Algorithmique
Correction

1) **Aiguiller**

Ecrire l'algorithme qui permet de lire 1 réel x , un caractère '+' ou '-' et un réel y, et qui effectue l'addition ou la soustraction suivant la valeur du caractère.

Algorithme à affiner :
- Lire les 2 nombres et l'opérateur
- Tenter de faire l'opération

```
soient x et y deux réels (données);
soit r réel (résultat) ;
soit c un caractère ;
lire(x) ;
lire(c) ;
lire(y) ;
si (c = '+')
  r <- x+y ;
  écrire(r) ;
sinon1
  si2 (c = '-')
    r <- x-y ;
    écrire(r) ;
  sinon2 erreur de saisie
    écrire(« erreur ») ;
  fin-si2
fin-si1
```

2) Enumérer

Ecrire l'algorithme qui lit un entier n et affiche les nombres entiers de 1 à n

Algorithme à affiner :
- Lire la valeur de n
- Générer toutes les valeurs à partir de 1 jusqu'à n

```
soit n, i deux entiers ;
lire(n) ;
i <- 1 ;
tant-que (i ≤ n)
  écrire(i) ;
  i <- i+1 ;
fin tant-que
```

qui affiche les nombres de n à 1

```
soit n, i deux entiers ;
lire(n) ;
i <- n ;
tant-que (0 < i)
  écrire(i) ;
  i <- i-1 ;
fin tant-que
```

3) Compter

Ecrire l'algorithme qui lit une phrase terminée par un point caractère par caractère et qui compte le nombre de 'e' dans cette phrase.

Algorithme à affiner :
- mettre un compteur à 0
- observer séquentiellement tous les caractères de la phrase
  incrémenter le compteur lorsque l'on observe un 'e'
- afficher le compteur

```
soit cpt un entier ;
soit c un caractère ;
cpt <- 0 ;
lire(c) ;
tant-que (c ≠ '.')
  si (c = 'e')
    cpt <- cpt+1;
  fin-si
  lire(c) ;
fin tant-que
ecrire(cpt) ;
```

4) Accumuler

Ecrire l'algorithme qui lit une suite d'entiers positifs terminée par -1 et trouve la moyenne de cette suite.

Algorithme à affiner :
- mettre un compteur à 0
- mettre un accumulateur à 0
- lire la suite de nombres en les comptant et en les cumulant dans l'accumulateur
- calculer la moyenne

```
soit x, s, cpt, moyenne quatre réels ;
s <- 0 ;
cpt <- 0 ;
lire(x) ;
tant-que (x ≠ -1)
  s <- s+x;
  cpt <- cpt+1;
```

```
  lire(x) ;
fin tant-que
moyenne <- s/cpt;
ecrire(moyenne) ;
```

5) Plus petite valeur

Ecrire l'algorithme qui lit une suite d'entiers positifs terminée par -1 et trouve le plus petit élément de la suite.

Algorithme à affiner :
- lire le 1er élément de la suite
- considérer que c'est le minimum provisoire de ce qui a été observé
- comparer séquentiellement les suivants pour retenir le plus petit
- afficher le minimum

```
soit x, min deux réels ;
lire(x) ;
min <- x ;            Le premier elt est le plus petit observé
lire(x) ;             on lit le suivant (qui peut-être la marque de fin !)
tant-que (x ≠ -1)     on répète la comparaison tant qu'on n'a pas atteint la fin
  si (x < min)
    min <- x ;
  fin-si
  lire(x) ;           on lit le suivant (qui peut-être la marque de fin !)
fin tant-que
ecrire(min) ;
```