Saisir un nombre ou bien le nombre d'éléments d'un tableau

```
fonction SaisirSimple():entier
début
   écrire("Saisir un entier:")
   lire(n)
   retourner (n)
fin

fonction SaisirAvecCondition1():entier
début
    répéter
        écrire("Saisir un entier positif:")
        lire(n)
    jusqu'à  n > 0
    retourner n
fin

fonction SaisirAvecCondition2() : entier
début
    écrire("Saisir un entier positif:")
    lire(n)
    tant que non( n>0) faire
        écrire("Saisir un entier positif:")
        lire(n)

    fin_tant_que
    retourner n
fin

fonction SaisirAvecCondition3():entier
début
    répéter
        écrire("Saisir un entier entre 20 et 500:")
        lire(n)
    jusqu'à  n ≥  20 Et n ≤ 500
    retourner n
fin
```

```
fonction SaisirAvecCondition4() : entier
début
    écrire("Saisir un entier entre 20 et 500:")
    lire(n)
    tant que non( n ≥ 20 Et n ≤ 500) faire
        écrire("Saisir un entier entre 20 et 500:")
        lire(n)

    fin_tant_que
    retourner n
fin
```

Remplissage d'un Tableau

```
procédure RemplirTableauSimple(@T:Tab,taille:entier)
début
    pour i de 0 à taille-1 faire
        écrire("Entrer une valeur :")
        lire(T[i])
    fin_pour
fin

procédure RemplirTableauCondition1(@T:Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        répéter
            écrire("Entrer une valeur entre 5 et 10 :")
            lire(T[i])
        jusqu'à  ( T[i] ≥ 5 Et T[i] ≤ 10 )

    fin_pour
fin
procédure RemplirTableauCondition2(@T:Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        écrire("Entrer une valeur entre 5 et 10 :")
        lire(T[i])
        tant que non ( T[i] ≥ 5 Et T[i] ≤ 10 ) faire
            écrire("Entrer une valeur entre 5 et 10 :")
            lire(T[i])

        fin_tant_que

    fin_pour
fin
```

```
procédure RemplirTableauCondition3(@T:Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        répéter
            écrire("Entrer une paire entre 5 et 35 :")
            lire(T[i])
        jusqu'à  ( T[i] ≥ 5 Et T[i] ≤ 35 ET T[i] mod 2 ==0 )

    fin_pour
fin
```

Affichage d'un Tableau

```
procédure AffichageTableauSimple(T:Tab, taille:entier)
début
    pour i de 0 à taille-1 faire
        écrire("T[", i, "]=",T[i])
    fin_pour
fin

procédure AffichageTableauCondition1(T: Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        si (T[i]> 0) alors
            écrire("T[", i, "]=",T[i])
        fin_si
    fin_pour
fin

procédure AffichageTableauCondition2(T: Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        si (T[i] mod 7 = 0) alors
            écrire("T[", i, "]=",T[i])
        fin_si
    fin_pour
fin
procédure AffichageTableauCondition3(T: Tab, taille: entier)
début
    pour i de 0 à taille-1 faire
        si (T[i] ≥ 3 Et T[i] ≤ 10) alors
            écrire("T[", i, "]=",T[i])
        fin_si
    fin_pour
fin
```

Manipulation d'un Tableau

```
fonction RechercheMaxTab(T: Tab, taille: entier) : entier
début
    max <-- T[0]
    pour i de 1 à taille-1 faire
        si max < T[i] alors
            max <-- T[i]
        fin_si
    fin_pour
    retourner max
fin

fonction RechercheMinTab(T: Tab, taille: entier) : réel
début
    min <-- T[0]
    pour i de 1 à taille-1 faire
        si min > T[i] alors
            min <-- T[i]
        fin_si
    fin_pour
    retourner min
fin
```

```
fonction RechercheElementTab1(T: Tab, taille: entier, elt: réel) : entier
début
   i <-- 0
   trouve <-- faux
   répéter
      si T[i] = elt alors
         trouve <-- vrai
      fin_si
      i <-- i+1
   jusqu'à  (i <= taille-1 Et trouve = faux)
   retourner i
fin
```

```
fonction RechercheElementTab2(T: Tab, taille: entier, elt: réel) : entier
début
   i <-- 0
   trouve <-- faux
   tant que non (i <= taille-1 Et trouve = faux) faire
      si T[i] = elt alors
         trouve <-- vrai
      fin_si
      i <-- i+1
   fin_tant_que
   retourner i
fin

fonction NombreOccurenceElementTab(T: Tab, taille: entier, elt: réel) : entier
début
   nb <-- 0
   pour i de 0 à taille-1 faire
      si T[i] = elt alors
         nb <-- nb+1
      fin_si
   fin_pour
   retourner nb
fin

fonction SommeElementsTab1(T: Tab, taille: entier) : réel
début
    somme <--0
    pour i de 0 à taille-1 faire
       somme <-- somme +T[i]
    fin_pour
    retourner somme
fin
```

```
fonction ProduitElementsTab2(T: Tab, taille: entier) : entier
début
    prod <-- 1
    pour i de 0 à taille-1 faire
        si (T[i] <> 0) alors
            prod <-- prod * T[i]
        fin_si
    fin_pour
    retourner prod
fin
```