1. Calcul de la moyenne et du minimum des éléments d'un tableau.

```
Moyenne (T, N) {
  somme <- 0;
  Pour i <- 1 a N Faire
    somme <- somme + T[i];
  moyenne <- somme / N;
  retourner moyenne;
}
```

2. Calcul du nombre d'occurences d'un élément donné dans un tableau.

```
Nb_occurences (T, N, X) {
  nb_occ <- 0;
  Pour i <- 1 a N Faire
   Si T[i] = X Alors
      nb_occ <- nb_occ + 1;
   retourner (nb_occ);
}

Minimum (T, N) {
  min <- T[1];
  Pour i <- 2 a N Faire
    Si T[i] < min  Alors {
      min <- T[i];
      pos_min <- i;
    }
  retourner (min, pos_min);
}
```

3. Écrire un algorithme qui teste si un tableau est trié.

```
Est_trie (T, N) {
  i <- 1;
  Tant que i < N ET T[i] <= T[i+1] Faire
     i <- i + 1;
  est_trie <- (i = N);
  retourner est_trie;
}
```

4. Écrire un algorithme qui teste si deux tableaux sont identiques.

```
Sont_identiques (T1, T2, N) {
  i <- 1;
  Tant que (i <= N) ET (T1[i] = T2[i]) Faire
   i <- i + 1;
  sont_identiques <- (i = N + 1);
  retourner sont_identiques;
}
```

5. Calcul du produit scalaire de deux vecteurs réels u et v de dimension n.

$$u.v = \sum_{i=1}^{i=n} u_i v_i$$

```
Produit_scalaire (u, v, n) {
  prod_scalaire <- 0;
  Pour i <- 1 a n Faire
    prod_scalaire <- prod_scalaire + u[i] * v[i];
  retourner prod_scalaire;
}
```

6. Décalage des éléments d'un tableau. Exemple :

   Tableau initial :

D	E	C	A	L	A	G	E

   Tableau modifié (décalage à gauche) :

E	C	A	L	A	G	E	D

```
Decalage_gauche (T, N) {
  tmp <- T[1];
  Pour i <- 1 a N-1 Faire
    T[i] <- T[i+1];
  T[N] <- tmp;
}
```

7. Calcul du produit de deux matrices carrées réelles $A = (a_{ij})$ et $B = (b_{ij})$ de dimension n.

$$c_{ij} = \sum_{k=1}^{k=n} a_{ik} b_{kj}$$

```
Produit_matriciel (a, b, n) {
  Pour i <- 1 a n Faire
    Pour j de 1 a n Faire {
      c[i][j] <- 0;
      Pour k de 1 a n Faire
         c[i][j] <- c[i][j] + a[i][k] * b[k][j];
    }
  retourner c;
}
```

8. Soit un tableau T avec $T(i) \in \{0, 1\}$. Écrire un algorithme qui retourne la position i dans le tableau telle que $T[i]$ est le début de la plus longue suite consécutive de zéros.

```
def pos_suite_0 (t):
  pos = -1
  lmax = 0
  suite = 0
  for i in range(0,len(t)):
    if t[i] == 0:
      if not suite:
        lg = 0
        suite = 1
      lg += 1
    else: # t[i] = 1
      if suite:
        suite = 0
        if lg > lmax:
          lmax = lg
          pos = i - lg
  if suite and lg > lmax:
     pos = i - lg + 1
  return pos
# O(len(t))
```

9. Écrire un algorithme qui calcule le plus grand écart dans un tableau (l'écart est la valeur absolue de la différence de deux éléments).

```
def plus_grand_ecart (t):
  min = t[0]
  max = t[0]
  for i in range(1,len(t)):
    if t[i] > max:
      max = t[i]
    else:
      if t[i] < min:
        min = t[i]
  return max - min
# O(len(t))
```