Algorithmes de mise à jour :

# 1.

**Algorithme** Inser\_Apres\_Valeur

**Constante** Max=100

**Type**Tab=tableau [1..max] d’entier

**Variable** A=Tab trouve=Booléen

Tail, i, val, elem: entire

**Procedure** rech\_seq(T:Tab,t,x:entire,var trv,ind:entire)

**Debut**

Ecrire(‘Donnez une taille à votre tableau’)

Lire(tail)

Pour i allant de 1 à tail faire

1. **DP**
2. lire(A[i])
3. **FP**

Ecire(‘Donnez la valeur après laquelle vous voulez insérer votre élément’)

Lire(val)

Rech\_seq(A,tail,val,trouve,indice)

Si (trouve) alors

1. **DSI**

Ecrire(‘Donnez l’élément à insérer’)

Lire(‘elem’)

A[tail+1]🡨A[indice+1]

A[indice+1]🡨elem

Tail🡨tail+1

Pour i allant de 1 à tail faire ecrire(A[i])

**FSI**

Sinon ecrire(‘Cet élément n’existe pas dans le tableau’)

**Fin.**

# 2.

**Algorithme** Inser\_pos

**Constante** max=100

**Type** Tab1d=tableau[1..max] d’entier

**Variable** v : Tab1d

i, tai, elem : entier

**Debut**

Lire(v,tai)

Lire(p) //position

Lire(elem)

Si p>max alors Ecrire(‘Erreur’)

Sinon

1. **DSINON**
2. Si p>tail alors V[tai+1]
3. Sinon
   1. **DSINON**
   2. pour i allant de tai à p (pas -1)
      1. **DP**
      2. V[i+1]🡨V[i]
      3. **FP**
   3. V[p]🡨elem
   4. tai🡨tail+1
   5. **FSINON**
4. **FSINON**

**FIN.**

# 3. Modifications :

- Recherche dichotomique.

- Le nombre d’itérations diminue.

## Pour le décalage :

Pour i allant de tail à (indice+1) (pas -1) faire T[i+1]🡨t[i]

# 4.

**Algorithme** sup\_logique

**Constante** max=100

**Type** Tab=Enregistrement

Val : entier

Sup : booleen

Tabsup=tableau[1..max] de Tab

**Variable** i, n, val, taille

T1 : TabSup

**Debut**

Lire (taille)

Pour i allant de 1 à taille faire

1. **DP**
2. Lire(T1[i].val)
3. T1[i].Sup🡨faux
4. lire(v)
5. rech\_seq(T1.val,taille,v,trouve,n)
6. Si trouve alors
   1. **DSI**
   2. T1[n].Sup🡨vrai
   3. **FSI**

**FIN.**

# 5.

**Algorithme** Sup\_phys

**Type** tab1d=tableau [1..100] d’entier

**Variable** taille, i, val

T : tab1d

**Debut**

Lire(taille)

Pour i allant de 1 à taille faire lire (T[i])

Lire(val)

Rech\_dicho (t, taille, val, troiuve, indice)

Si (trouve=vrai)

1. **DSI**
2. k🡨indice+1
3. pour j allant de k à taille faire
   1. **DP**
   2. T[j-1]🡨T[j]
   3. **FP**
4. **FSI**

Sinon ecrire(‘l’élément recherché n’existe pas’)

**FIN.**