☒ **Algorithme du Programme principal** :

0) Début Ex27Mai2013_8h
1) PROC Saisie(n)
2) PROC Remplir(M,n)
3) PROC Remplir2(S,n)
4) PROC Tri(S,M, n)
5) PROC Affichage(M, n)
6) Fin Ex27Mai2013_8h

☒ **Algorithme de la procédure Saisie** :

0) DEF PROC Saisie (var n :entier)
1) [ ] **Répéter**
Ecrire("saisir la taille d'un tableau=")
Lire(n)
Jusqu'à (n dans [5..10])
2) Fin Saisie

☒ **Algorithme de la procédure Remplir** :

0) DEF PROC Remplir **(var M :tab; n :entier)**
1) [ ] **Pour** i de 1 à n **faire**
Répéter
Ecrire("M[",i,"]="),Lire(M[i])
[ok←vrai,j←0]répéter
j←j+1
si (non (M[i][j] dans ["0"..."9"]) alors
ok←Faux
finsi
jusqu'à (ok=Faux) ou (j=long(ch))
jusqu'à (long(M[i]=8) et (ok=vrai)
FinPour
2) Fin Remplir

☒ **Algorithme de la procédure Remplir2** :

3) DEF PROC Remplir2**(var S :tab; n :entier)**
4) [ ] **Pour** i de 1 à n **faire**
Répéter
Ecrire("S[",i,"]="),Lire(S[i])
jusqu'à (S[i] dans [20..120])
FinPour
5) Fin Remplir

☒ **Algorithme de la procédure Tri**: (1ére méthode :tri_sélection)

0) DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)
1) [ ] **Pour i de 1 à n-1 faire**
[pos_max←i] **Pour** j **d**e i+1 **à** n **faire**
[ ] **Si** (M[j] >M[pos_max])**Alors**
pos_max←j
Finsi
FinPour
[ ] **Si** (pos_max ≠i)**Alors**
aux←s[i]
s[i]←[pos_max]
s[pos_max] ←aux
aux2←m[i]
m[i] ←m[pos_max]

```
            m[pos_max] ←aux2
        Finsi
    FinPour
2) FIN Tri
```

☒ **Algorithme de la procédure Tri**: (2éme méthode :tri_bulles)

```
0) DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)
1) Répéter
   [ Echange←faux] Pour i de 1 à n-1 faire
                        [ ]  Si ( S[i] < S[i+1])Alors
                                aux←s[i]
                                s[i] ←s[i+1]
                                s[i+1] ←aux
                                aux2←m[i]
                                m[i] ←m[i+1]
                                m[i+1] ←aux2
                               Echange ← vrai
                           FinSi
                      FinPour
       n←n-1
   Jusqu'à (Echange = Faux) ou (n=1)
2) Fin Tri
```

☒ **Algorithme de la procédure Tri**: (3éme méthode :tri_insertion)

```
0) DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)
1) [ ] Pour i de 2 à n faire
     [x←S[i], y←M[i],j←i ]
   Tant que ( j >1) et ( S[j-1] < x ) faire
        S[j ]← S[j-1]
        M[j ]← M[j-1]
        j ← j - 1
   FinTantQue
   S[j ] ←x
   M[j ] ←y
   FinPour
2) Fin Tri_Insertion
```

☒ **Algorithme de la procédure Affichage**:

```
0) DEF PROC Affichage (M :tab2 ;n :entier)
1) [ ] Pour i de 1 à arrondi(n*0.25) faire
        Ecrire(M[i]," ")
   FinPour
2) Fin Affichage
```