* **Algorithme du Programme principal** :

1. **Début Ex27Mai2013\_8h**
2. **PROC Saisie(n)**
3. **PROC Remplir(M,n)**
4. **PROC Remplir2(S,n)**
5. **PROC Tri(S,M, n)**
6. **PROC Affichage(M, n)**
7. **Fin Ex27Mai2013\_8h**

* **Algorithme de la procédure Saisie** :

1. **DEF PROC Saisie (var n :entier)**
2. [ ] **Répéter**

Ecrire("saisir la taille d'un tableau=")

Lire(n)

**Jusqu’à (n dans [5..10])**

1. **Fin Saisie**

* **Algorithme de la procédure Remplir** :

1. **DEF PROC** Remplir **( var M :tab; n :entier)**
2. [ ] **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**

1. **Fin Remplir**

* **Algorithme de la procédure Remplir2** :

1. **DEF PROC** Remplir2**( var S :tab; n :entier)**
2. [ ] **Pour** i de 1 à n **faire**

**Répéter**

Ecrire("S[",i,"]="),Lire(S[i])

**jusqu’à (S[i] dans [20..120])**

**FinPour**

1. **Fin Remplir**

* **Algorithme de la procédure Tri**: (1ére méthode :tri\_sélection)

1. **DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)**
2. [ ] **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**

1. **FIN Tri**

* **Algorithme de la procédure Tri**: (2éme méthode :tri\_bulles)

1. **DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)**
2. **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)

1. **Fin Tri**

* **Algorithme de la procédure Tri**: (3éme méthode :tri\_insertion)

1. **DEF PROC Tri(Var S:tab; Var M :tab2 ;n :entier)**
2. [ ] **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**

1. **Fin Tri\_Insertion**

* **Algorithme de la procédure Affichage**:

1. **DEF PROC** Affichage **(M :tab2 ;n :entier)**
2. [ ] **Pour** i de 1 à **arrondi(n\*0.25)** **faire**

Ecrire(M[i]," ")

**FinPour**

1. **Fin Affichage**