Corrigé

1. Definir la structure de la liste L

Type
Pliste=^ListeH ;(1)
ListeH=Enregistrement
Val :entier ;
Suiv :Pliste ;
Fin ;
Vliste=^ListeV; (0.5)
ListeV=Enregistrement
Hsuiv :Pliste ;
Vsuiv :Vliste ;
Fin ;

2. Transformer L en LN(2.5)

Procedure **Liste_Lineaire (E/S L :Vliste ; S/ LN : Pliste);**
Var P, Q:Pliste; LV: Vliste;
Debut
{récupérer la tête de la liste Horizontale LH}
LN← L^.Hsuiv ;{LN=tête de la nouvelle Liste}
LV←L^.Vsuiv ;{LV pour parcourir la liste Verticale}
Q←LN ;
{Q va pointer sur le dernier élément de la liste Horizontale LH}
Tantque(LV<>Nil) **faire**
Tantque(Q^.suiv<>Nil) **Faire**
Q←Q^.suiv ;
Fait ;
P←LV^.Hsuiv ;{P=la tête de la LH suivante pour la relier à l'élément pointé par Q}
Q^.suiv←P ;
Liberer (L) ;{libérer l'élément de la liste verticale}

L←LV ;{Nouvelle tête de liste verticale à libérer}

LV←LV^.Vsuiv ;{avancer dans la Liste Verticale}

Fait ;

Fin ;

3. Recherche récursive d'une valeur V donnée

Fonction Rech_rec(E/SPt :Pliste ;V: entier) :booleen ;(2)
Debut
SI Pt= NIL alors Rech_rec :=faux
Sinon si Pt.Val=V
ALORS Rech_rec :=Vrai
SINON RECH_REC :=Rech_rec (Pt^.Suiv,V)
FIN ;

Corrigé

4. Suppression de la Kiemecellule

Procédure Supprimer_K (E/S LN : Pliste ; E K : entier) ;(2.5)
Var
S, P : Pliste ;

Début
S ← LN ;
Tantque (K>1) **faire**
P ← S ;
S ←S^. Suivant ;
K ← K-1 ;
Fait ;
LN ← S^. Suivant ;
P^. Suivant ← LN ;
Libérer (S) ;
Fin ;

5. Calcule de la complexité(1)

Début
S ← LN ; initialisation : 1 fois
Tantque (K>1) **faire** itérations : (K-1) fois + 1 test de sortie de la boucle
P ← S ; affectation : (K-1) fois
S ←S^. Suivant ; affectation (K-1) fois
K ← K-1 ; soustraction + affectation : 2*(K-1) fois
Fait ;
LN ← S^. Suivant ; affectation : 1 fois
P^. Suivant ← LN ; affectation : 1 fois
Libérer (S) ; affectation : 1 fois
Fin.

C_tantque(K)= (1*K)+(K-1)+(K-1)+ 2*(K-1)=5K-4
C_algo (K) = 1+5K-4+3=5K

6. Suppression circulaire(2.5)

Procédure Supprimer_tt_K (E/S LN : Pliste ; E K : entier) ;

Début
Tantque (LN<>LN^. Suivant) **faire**
Supprimer_K (LN,K) ;
Fin Tantque ;
Liberer (LN) ;
LN← Nil ;
Fin ;

Corrigé

Exercice 2 :

Type

Mot=chaine de caractères [25](0.5)

Fich=fichier de mot(0.5)

File =^Element(0.5)

Element = enregistrement

Pos :entier

Suiv :file

Fin ;

Pile =^Nœud(1,5)

Nœud = enregistrement

Awal :Mot

Posit :File

Suiv :Pile

Fin ;

Var

T :Fich ;

ProcédurePosMOT(E/S FM :Fich, ; E Mt : mot ; S F,Q :File) (2,50)

```
Var
        X :mot ;
        i :entier ;
Debut
        Reset(FM)
        i:=1;
        InitFile(F,Q)
        Tant que not(Eof(FM))
        Faire lire(FM,X)
        Si X=Mt alors
        Enfiler(F,Q,i) ;
        i :=i+1
        Fsi
        Fait
        Close(FM)
Fin;
```

Procedure ExtraireMot(E/S FMot :Fich, ; S P:pile)

Var(2,5)

```
X:mot ;
G:Fich;
E :Noeud;
F,Q:file ;
Debut
   Initpile(P)
   Assigne(G,"inter.dat")
   Copy(Fmot,G) {* dupliquer le fichier Fmot*}
   Reset(Fmot);
   Tant que not(EOf(Fmot))
   Faire Lire(Fmot,X)
        Si RechMotPile(P,X) = faux
        Alors PosMOT(G,X, F,Q )
        E.Mot := ;
        E.Posit :=F ;
        Empiler(P,E)
        Fsi
   Fait ;
Fin.
```