Corrigé Type : Algorithmique Rattrapage S2

Exercice 01 :

1- Proc retourne la cellule située au milieu d'une liste linéaire chainée si la liste contient un nombre impair de cellules, ou l'une des deux cellules du milieu si le nombre de cellules de la liste est pair. (2)

2- Version de Proc avec un seul parcours :

Procédure Proc2 (E Tête : liste ; S CCourant : liste); (2)

Var Courant, CCourant : liste;

Début

Courant ← Tête;

CCourant ← Tête;

Tant que (CCourant <> NiL) et (CCourant^.Suivant <> NiL) faire

Courant ← Courant^.Suivant;

CCourant ← CCourant^.Suivant^.Suivant;

Fin Tant que;

Fin;

3- Les Complexités algorithmiques de Proc et Proc2 : (2)

Sachant $P = (\frac{n}{2} + 1)$;

$C_{Proc} = 2 + (n+1) + 3n + 4 + 1 + P + 1 + P = 9 + 3n + n + (\frac{n}{2} + 1) + (\frac{n}{2} + 1)$

$= 9 + 4n + n + 2 = \boxed{5n + 11} \Rightarrow$ Un ordre de grandeur $O(n)$

$C_{Proc2} = 2 + 3(\frac{n}{2} + 1) + 2(\frac{n}{2}) = \frac{3n}{2} + \frac{2n}{2} + 2 + 3 = \boxed{\frac{5n}{2} + 5} \Rightarrow$ Un ordre de grandeur $O(n)$

4- Les deux Procédures sont de la même classe de complexité (Complexité linéaire). Cependant, Proc2 est meilleure que Proc en terme de complexité algorithmique. (2)

5- La version récursive de Proc2 : (2)

Procédure Proc2_recursive (E Courant, CCourant : Liste);

Début

①

```
Si (CCourant <> NIL) et (CCourant^.Suivant <> NIL) alors
    Ecrire (Courant^.Info);
Sinon
    Proc2_Recursive (Courant^.Suivant, CCourant^.Suivant^.Suivant);
Fin Si;
Fin;
```

Exercice 02 :

1 -

```
Algorithme Exo2_1;
Var F1, F2, G: Fichiers d'entiers;
    x, y: entier;
Début
    Associer (F1, 'Fichier1.dat');
    Associer (F2, 'fichier2.dat');
    Associer (G, 'Fichier3.dat');
    Relire (F1);
    Réécrire (G);
    Tant que (Non FDF(F1)) faire
        Lire (F1, x);
        Ecrire (G, x);
        Relire (F2);
        Tant que (Non FDF(F2)) faire
            Lire (F2, y);
            Si (y mod x = 0) alors
                Ecrire (G, y);
            Fin Si;
        Fin Tant que;
```

0,25

1

1

```
        Fermer (F2);
    Fin Tant que;
    Fermer (F1);
    Fermer (G);
Fin.
```

2|

```
Algorithme Exo2_2;
Type  Liste = ^Cellule;
      Cellule = enregistrement
          Valeur : entier;
          Repetition : entier;
          Suivant : Liste;
      Fin;
Var
    x : entier;
    Courant, Tête, Nouveau, Queue : Liste;

Fonction Existe (Tête : Liste; V : entier) : Booléen;
  Var
      Courant : Liste;
      Trouve : Booléen;
  Début
      Courant ← Tête;
      Trouve ← Faux;
      Tant que (Courant <> NiL) et (Non Trouve) faire
          Si (Courant^.Valeur = V) alors
              Trouve ← Vrai;
```

```
        Sinon
            Courant ← Courant^.Suivant;
        Fin Si;
    Fin Tant que;
    Existe ← Trouve;
Fin;
```

```
Procédure Chercher (E Tête : liste; E V : entier; S Courant : liste);
Var Courant : liste;
Debut
    Courant ← Tête;
    Tant que (Courant <> NIL) et (Courant^.Valeur <> V) faire
        Courant ← Courant^.Suivant;
    Fin Tant que;
Fin;
```

```
Debut
    Allouer (Tête);
    Associer (G, 'Fichier3.dat');
    Reliré (G);
    Lire (G, x);
    Tête^.Valeur ← x;
    Tête^.Repetition ← 1;
    Tête^.Suivant ← NIL;
    Queue ← Tête;
    Tant que (Non FDF(G)) faire
        Lire (G, x);
        Si (non Existe (Tête, x)) alors
            Allouer (Nouveau);
            Nouveau^.Valeur ← x;
            Nouveau^.Repetition ← 1;
            Nouveau^.Suivant ← NIL;
            Queue ← Nouveau;
        Sinon
            Chercher (Tête, x, Courant);
            Courant^.Repetition ← Courant^.Repetition + 1
        Fin Si;
    Fin Tant que;
    Queue^.Suivant ← Tête; Courant ← NIL;
Fin. Nouveau ← NIL;
```