امتحان في مادة الخوارزميات 2

الحل النموذجي

السداسي الثاني: 2026/2025

Exercice 01 - 10 PTS

Q1) FCT(12 , 0) = 21 (0.50) FCT(159 , 0) = 951 (0.50)

Q2) La fonction **FCT** inverse les chiffres d'un nombre entier (0.50)

Q3) 1.25 Pts

```
Fonction Puissance(A, k : Entier) : Entier        (0.25)
   VAR      i, P : Entier
Début
   P ← 1                                          (0.25)
   Pour i ← 1 à k Faire                           (0.25)
      P ← P * A                                   (0.25)
   FinPour
   Puissance ← P                                  (0.25)
Fin
```

Q4) 1.25 Pts

```
Fonction nbCH(X : Entier) : Entier                (0.25)
Début
   Si (X < 10) Alors                              (0.25)
      nbCH ← 1                                    (0.25)
   Sinon
      nbCH ← 1 + nbCH(X DIV 10)                   (0.50)
   FinSi
Fin
```

Q5) 1.50 Pts

```
Fonction Inverse(X : Entier) : Entier
  VAR   Chiffre, POS : Entier
Début
  Si (X < 10) Alors                               (0.25)
        Inverse ← X                               (0.25)
  Sinon
        Chiffre ← X MOD 10                        (0.25)
        POS ← Puissance (10, nbCH(X) - 1)         (0.25)
        Inverse ← Chiffre * POS + Inverse(X DIV 10)   (0.50)
  FinSi
Fin
```

(1.00Pt)

Q6) 0.75 Pts

```
Fonction estPalindrome(X : Entier) : Booléen      (0.25)

Début
   Si (X = Inverse(X)) Alors
      estPalindrome ← Vrai
   Sinon
      estPalindrome ← Faux
   FinSi
Fin
```

(0.50)

Solution 2

```
Fonction estPalindrome(X : Entier) : Booléen      (0.25)
Début
  estPalindrome ← (X = Inverse(X))                (0.50)
Fin
```

Q7)

2.00 Pts

```
Procédure ordrePalind(X : Entier, VAR K : Entier, VAR XPal : Entier)        0.25
Début
    K ← 0                                                                   0.25
    TantQue (NON estPalindrome(X)) Faire                                    0.75
        X ← X + Inverse(X)                                                  0.25
        K ← K + 1                                                           0.25
    FinTQ
    XPal ← X                                                                0.25
Fin
```

Solution 2 :

2.00 Pts

```
Procédure ordrePalind(X : Entier, VAR K : Entier, VAR XPal : Entier)        0.25
    VAR     Palindrom : Booléen
Début
    K ← 0
    Palindrom ← Faux                                                        0.25
    TantQue (Non Palindrom) Faire                                           0.25
        Si estPalindrome(X) Alors                                           0.25
            Palindrom ← Vrai                                                0.25
        Sinon
            X ← X + Inverse(X)                                              0.25
            K ← K + 1                                                       0.25
        FinSi
    FinTQ
    XPal ← X                                                                0.25
Fin
```

Solution 3 : Version RECURSIVE

2.00 Pts

```
Procédure ordrePalind(X : Entier, VAR K : Entier, VAR XPal : Entier)        0.25
Début
    Si estPalindrome(X) Alors                                               0.50
        K ← 0                                                               0.25
        XPal ← X                                                            0.25
    Sinon
        ordrePalind(X + Inverse(X), K, XPal)                                0.50
        K ← K + 1                                                           0.25
    FinSi
Fin
```

Q8) 1.75 Pts

```
Algorithme Recherche
   VAR      X, K, XPal, Cpt : Entier
   Fonction Puissance(A, k : Entier) : Entier
   Fonction nbCH(X : Entier) : Entier
   Fonction Inverse(X : Entier) : Entier
   Fonction estPalindrome(X : Entier) : Booléen
   Procédure ordrePalind(X : Entier, VAR K : Entier, VAR XPal : Entier)
Début
   X ← 1000
   Cpt ← 0
   TantQue (Cpt < 3) Faire
      ordrePalind(X, K, XPal)
      Si (K > 10) Alors
         Écrire(X)
         Cpt ← Cpt + 1
      FinSi
      X ← X + 1
   FinTQ
Fin
```

0.25 (Fonctions)
0.25 (X ← 1000, Cpt ← 0)
0.25 (TantQue)
0.25 (ordrePalind)
0.50 (Si … FinSi)
0.25 (X ← X + 1)

Exercice 02 - 10 PTS

Q1) (0.5pt) Type Liste = ^Element **(0.25 pt)**

Element = **Enregistrement** **(0.25 pt)**
- Val : Entier
- Suiv : Liste

Fin

Q2) (1.5 pts) Procédure InsererQueue(**Var** L : Liste ; N : Entier) **(0.25 pt)** // Itérative

Var P, Dernier : Liste

Début

- Allouer(P)
- P^.Val ← N
- P^.Suiv ← Nil

(0.50 pt)

- **Si** (L = Nil) **Alors**
  - L ← P

(0.25 pt)

- **Sinon**
  - Dernier ← L
  - **TQ** (Dernier^.Suiv ≠ Nil) **Faire**
    - Dernier ← Dernier^.Suiv
  - **FTQ**
  - Dernier^.Suiv ← P

(0.50 pt)

- **FSi**

Fin

Procédure InsererQueue(**Var** L : Liste ; N : Entier) **(0.25 pt)** // Récursive

Var P : Liste

Début

Si (L = Nil) **Alors** **(0.25 pt)**

Allouer(P)

P^.Val ← N **(0.50 pt)**

P^.Suiv ← Nil

L ← P

Sinon

InsererQueue(L^.Suiv, N) **(0.50 pt)**

FSi

Fin

// Solution 2

Allouer(L)

L^.Val ← N

L^.Suiv ← Nil

Q3) (2 pt) Procédure CreerListe(**Var** L : Liste) **(0.25 pt)** // Version récursive

Var N : Entier

Début

Ecrire(" Donner un entier positif N. Une valeur négative arrête la saisie ")

Lire(N) **(0.25 pt)**

Si (N ≥ 0) **Alors** **(0.25 pt)**

InsererQueue(L, N) **(0.25 pt)**

CreerListe(L)**(1 pt)**

FSi

Fin

(1.25 pt) Procédure CreerListe(**Var** L : Liste) **(0.25 pt)** // Version itérative

Var N : Entier

Début

Répéter

Lire(N) **(0.25 pt)**

Si (N ≥ 0) **Alors** **(0.25 pt)**

InsererQueue(L, N) **(0.25 pt)**

FSi

Jusqu'à (N <0) **(0.25 pt)**

Fin

Lire(N) (0.25 pt)

TantQue (N>=0) faire (0.25 pt)

InsererQueue(L,N) (0.25 pt)

Lire(N) (0.25 pt)

FinTQ

Q4) (2.5 pts) Procédure CompresserListe(L1 : Liste ; **Var** L2 : Liste)

```
Var     N, Cpt : Entier
Début
    L2 ← Nil                                    (0.25 pt)
    Si ( L1 ≠ Nil ) Alors                       (0.25 pt)
        N ← L1^.Val                             // Initialisation de la première valeur
        Cpt ← 1                                 (0.25 pt)
        L1 ← L1^.Suiv                           (0.25 pt)
        TQ ( L1 ≠ Nil ) Faire                   (0.25 pt)
            Si ( N = L1^.Val ) Alors
                Cpt ← Cpt + 1                   (0.25 pt)
            Sinon
                InsererQueue( L2, N )
                InsererQueue( L2, Cpt )         (0.25 pt)
                N ← L1^.Val
                Cpt ← 1                         (0.25 pt)
            FSi
            L1 ← L1^.Suiv                       (0.25 pt)
        FTQ
        InsererQueue( L2, N )                   // Sauvegarder le dernier groupe
        InsererQueue( L2, Cpt )                 (0.25 pt)
    FSi
Fin
```

(2.5 pts) Procédure CompresserListe(L1 : Liste, VAR L2 : Liste)

```
    VAR    N, Cpt : Entier
Début
    L2 ← NIL                                    0.25
    TantQue (L1 ≠ NIL) Faire                    0.50
        N ← L1^.Val                             0.25
        Cpt ← 1
        L1 ← L1^.Suiv                           0.25
        TantQue (L1 ≠ NIL) ET (L1^.Val = N) Faire   0.50
            Cpt ← Cpt + 1                       0.25
            L1 ← L1^.Suiv                       0.25
        FinTQ
        InsererFin(L2, N)                       0.25
        InsererFin(L2, Cpt)
    FinTQ
Fin
```

Q5) (1 pt) Fonction Longueur(L : Liste) : Entier // Version itérative

Var Cpt : Entier

Début

Cpt ← 0 **(0.25 pt)**

TQ (L ≠ Nil) **Faire** **(0.25pt)**

Cpt ← Cpt + 1

L ← L^.Suiv **(0.25 pt)**

FTQ

Longueur ← Cpt **(0.25 pt)**

Fin

Fonction Longueur(L : Liste) : Entier // Version récursive

Début

Si (L = Nil) **Alors** **(0.25 pt)**

Longueur ← 0 **(0.25 pt)**

Sinon

Longueur ← 1 + Longueur(L^.Suiv) **(0.50 pt)**

FSi

Fin

Q6) (1 pt) Procédure SuppListe(**Var** L : Liste) // Version récursive **SEULEMENT**

Var P : Liste

Début

Si (L ≠ Nil) **Alors** **(0.25 pt)**

P ← L

L ← L^.Suiv **(0.25 pt)**

Libérer(P) **(0.25 pt)**

SuppListe(L) **(0.25 pt)**

FSi

Fin

Q7) (1.5 pts) Algorithme TestDeLaCompression

```
Type Liste = ^Element
     Element = Enregistrement        // Déclaration du type Liste
     |    Val : Entier
     |    Suiv : Liste
     Fin
Var  L1, L2 : Liste
Procédure InsererQueue( Var L : Liste ; N : Entier ) …
Procédure CreerListe( Var L : Liste ) …                                  // Rappel des procédures et fonctions
Procédure CompresserListe( L1 : Liste ; Var L2 : Liste ) …               (0.25 pt)
Fonction Longueur( L : Liste ) : Entier …
Procédure SuppListe( Var L : Liste ) …
Début
|    L1 ← Nil                                  // Si non incluse dans la définition de CreerListe(…)
|    CreerListe( L1 )          (0.25 pt)      // Création correcte de la liste L1
|    CompresserListe( L1, L2 )  (0.25 pt)
|    // Vérifier si l'opération de compression était utile ou pas en comparant les tailles des deux listes.
|    Si ( Longueur( L1 ) > Longueur( L2 ) ) Alors
|    |    Ecrire(" La compression de la liste L1 est utile ")
|    Sinon                                                                (0.50 pt)
|    |    Ecrire(" La compression de la liste L1 n'est pas utile ")
|    FSi
|    SuppListe( L1 )
|    SuppListe( L2 )          (0.25 pt)      // Suppression des deux listes.
Fin
```

****** Fin du corrigé-type ******