

Correction TD 04 : Les Piles et les files

Exercice 1 :

3.

Procédure Inverser(T: Tab; N: Entier)

var

P: Pile

i: Entier

Début

Init_Pile(&P)

Pour i de 1 à N Faire

Empiler (&P, T[i])

Fin pour

Pour i de 1 à N Faire

T[i] <- Dépiler (&P)

Fin pour

Fin

Exercice 2 :

1.

Procédure Inverser_Pile (P: *Pile)

var

P1, P2: Pile

Début

Init_Pile(&P1)

Init_Pile(&P2)

Tant que(vide(*P) = faux) Faire

Empiler (&P1, Dépiler(P))

Fin Tant que

Tant que (vide(P1) = faux) Faire

Empiler (&P2, Dépiler (&P1))

Fin Tant que

Tant que (vide(P2) = faux) Faire

Empiler (P, Dépiler (&P2))

Fin Tant que

Fin

2.

```
Procédure Copie_Pile(P: Pile)  
var  
    Q1, Q2: Pile  
Début  
    Init_Pile(Q1)  
    Init_Pile(Q2)  
  
    Tant que (vide(P) = faux) Faire  
        Empiler (&Q2, Dépiler(P))  
    Fin Tant que  
  
    Tant que (vide(Q2) = faux) Faire  
        Empiler (&Q1, Dépiler (&Q2))  
    Fin Tant que  
    Copie_Pile <- Q1  
  
Fin
```

3.

```
Procédure Ranger(T:Tab; N:Entier)  
var  
    P, Q: Pile  
    i:Entier  
Début  
    Init_Pile(P)  
    Init_Pile(Q)  
    Pour i de 1 à N Faire  
        Si (T[i] mod 2 = 0) alors  
            Empiler (&P, T[i])  
        Sinon  
            Empiler (&Q, T[i])  
        Fin Si  
    Fin Pour  
    i <- N  
    Tant que (vide(Q) = faux) Faire  
        T[i] <- Dépiler(&Q)  
        i <- i - 1  
    Fin Tant que  
  
    Tant que (vide(P) = faux) Faire  
        T[i] <- Dépiler(&P)  
        i <- i - 1  
    Fin Tant que  
  
Fin
```

Exercice 3 :

```
Procédure Trier_File (F: *File)  
var  
    P, Q: Pile  
    X: entier  
  
Début  
    Init_Pile(&P)  
    Init_Pile(&Q)  
  
    Tant que (Vide (*F) = faux) Faire  
        Empiler (&P, Défiler(F))  
    Fin Tant que  
  
    Tant que (Vide(P) = faux) Faire  
        x <- Dépiler (&P)  
        Tant que (Vide(Q)= faux ET Sommet(Q) < x) Faire  
            Empiler (&P, Dépiler (&Q))  
        Fin Tant que  
  
        Empiler (&Q, x)  
    Fin Tant que  
  
    Tant que (Vide(Q)= faux) Faire  
        Enfiler (F, Dépiler (&Q))  
    Fin Tant que  
  
Fin
```

Exercice 4:

```
Fonction evaluer (ch: chaîne): entier  
var  
    i, x, y, z: entier  
    P: pile  
  
Début  
    init_pile(&P)  
    Pour i de 1 à Long (ch) Faire  
        Selon ch[i] Faire  
            Cas '+':  
                x <- Dépiler (&P)  
                y <- Dépiler (&P)  
                z <- x + y  
                Empiler (&P, z)
```

```

        Cas '-':
            x <- Dépiler (&P)
            y <- Dépiler (&P)
            z <- y - x
            Empiler (&P, z)
        Cas '*':
            x <- Dépiler (&P)
            y <- Dépiler (&P)
            z <- x * y
            Empiler (&P, z)
        Cas '/':
            x <- Dépiler (&P)
            y <- Dépiler (&P)
            z <- y / x
            Empiler (&P, z)
        Sinon:
            Empiler (&P, ch[i]))
    Fin Selon
Fin Pour
evaluer <- Dépiler(&P)
Fin

```

Exercice 5 :

```

Algorithme Ranger
var
    P, Q, W : Piles
Début
    init_pile(&Q)
    init_pile(&W)

    Tant que (Vide(P) = faux) Faire
        Si (sommet(P) mod 2 = 1) Alors
            Empiler (&Q, Dépiler (&P))
        Sinon
            Empiler (&W, Dépiler (&P))
        Fin Si
    Fin Tant que

    Tant que (Vide(W) = faux) Faire
        Empiler (&Q, Dépiler (&W))
    Fin Tant que
Fin

```