Travaux Dirigés n°2

MAIN3
VOYLES Evan
October 13, 2021

Algorithm 1 Renvoi l'indice du plus petit élement d'un tableau d'entiers

Données d'entrée : T (tableau d'entiers), N (nombre entier, taille de T)
Données de sortie : indice_min (nombre entier)

1: $indice\_min \leftarrow$ -1
2: $min \leftarrow T_1$
3: **for** $i \leftarrow 1, N$ **do**
4: **if** $T_i < min$ **then**
5: $min \leftarrow T_i$
6: $indice\_min \leftarrow i$
7: **end if**
8: **end for**
9: **return** $indice\_min$

Algorithm 2 Nombre d'occurences

Données d'entrée : T (tableau d'entiers), n (nombre entier), P (nombre entier, taille de T)
Données de sortie : nombre_fois (nombre entier)

1: $nombre\_fois \leftarrow 0$
2: **for** $i \leftarrow 1, P$ **do**
3: **if** $T_i = n$ **then**
4: $nombre\_fois \leftarrow nombre\_fois + 1$
5: **end if**
6: **end for**
7: **return** $nombre\_fois$

Algorithm 3 Moyenne et ecart-type

Données d'entrée : T (tableau de doubles), N (nombre entier, taille de T)
Données de sortie : moyenne (double), ecart_type (double)

1: $moyenne \leftarrow 0$
2: $ecart\_type \leftarrow 0$
3: $sum \leftarrow 0$
4: $ecart\_carre \leftarrow 0$
5: **for** $i \leftarrow 1, N$ **do**
6: $sum \leftarrow sum + T_i$
7: **end for**
8: $moyenne \leftarrow sum/N$
9: **for** $i \leftarrow 1, N$ **do**
10: $ecart\_carre \leftarrow (T_i - moyenne)^2$
11: **end for**
12: $ecart\_type \leftarrow \sqrt{ecart\_carre/(N-1)}$
13: **return** $moyenne, ecart\_type$

Algorithm 4 Palindrome

Données d'entrée : T (tableau d'entiers), N (nombre entier, taille de T)
Données de sortie : est_palindrome (boolean)

1: **for** $i \leftarrow 1, \lfloor N/2 \rfloor$ **do**
2: **if** $T_i \neq T_{N-i+1}$ **then**
3: **return** $false$
4: **end if**
5: **end for**
6: **return** $true$

Algorithm 5 Remplacement des doublons

Données d'entrée : T (tableau d'entiers > 0), N (nombre entier, taille de T)
Données de sortie : T avec les doublons remplace par 0

1: $old\_value \leftarrow T_1$
2: **for** $i \leftarrow 2, N$ **do**
3: **if** $T_i = old\_value$ **then**
4: $T_i \leftarrow 0$
5: **else**
6: $old\_value \leftarrow T_i$
7: **end if**
8: **end for**
9: **return** T

Algorithm 7 Crible d'Érastosthène

Données d'entrée : T (tableau d'entiers > 0), N (nombre entier)
Données de sortie : T avec les éléments non-primes remplacés par -1

1: $T \leftarrow 2 : N$
2: **for** $i \leftarrow 2, \left\lfloor \sqrt{N} \right\rfloor$ **do**
3: **for** $j \leftarrow 1, N$ **do**
4: **if** $T_j \bmod i \equiv 0$ **then**
5: $T_j \leftarrow -1$
6: **end if**
7: **end for**
8: **end for**
9: **return** T

Travaux Dirigés n°3

Algorithm 1 Crible d'Érastosthène

Données d'entrée : T (tableau d'entiers > 0), N (nombre entier)
Données de sortie : T avec les éléments non-primes remplacés par -1

1: $T \leftarrow 2 : N$
2: **for** $i \leftarrow 2, \left\lfloor \sqrt{N} \right\rfloor$ **do**
3: **for** $j \leftarrow 1, N$ **do**
4: **if** $T_j \bmod i \equiv 0$ **then**
5: $T_j \leftarrow -1$
6: **end if**
7: **end for**
8: **end for**
9: **return** T

Travaux Dirigés n°4

Exercice 3. *On considère le programe suivant:*

```c
#include <stdio.h>
int a = 27;
int chose(int a);
int machin();

int chose(int a)
{
    return a+17+machin(); // a est une variable locale
}
int machin()
{
    return a; // a est une variable globale
}
int main()
{
    int a=1; // nouvelle declaration de a cache le 'a' de ligne 2
    a=chose(a); // a <- (1 + 17 + 27)
    printf("%d\n", a);
}
```

Affiche 45 a l'écran.

Exercice 4. *Définissez l'affichage produit par l'exécution du programme. Expliquez.*

```c
#include <stdio.h>

int nb = 3;
//============================================================

void mal_ecrit_1 (int nb) // nb est une variable locale
{
    printf ("nb mal_ecrit_1 = %d\n",nb++); // nb++ renvoie nb et s'incremente
    APRES
    {
        int nb = 14; // cette exo est deguelasse
        printf ("nb mal_ecrit_1 = %d\n",nb); // nb = 14
    }
    printf ("nb mal_ecrit_1 = %d\n",nb); // nb locale
}
//============================================================

int mal_ecrit_2 (int x, int y) // nb dans cette scope est globale
{
    printf ("nb mal_ecrit_2 = %d\n",x + nb);
    printf ("nb mal_ecrit_2 = %d\n",y + nb);
    return (nb *= 0); // nb <- 0
}
//============================================================
```

```c

int main(void)
{
    int x = 3;
    int y = 6;
    printf ("nb main = %d\n",nb); // nb globale est initialise a 3
    mal_ecrit_1(nb); // ne modifie pas du tout le nb globale
    printf ("nb main = %d\n",nb);
    printf("resultat de mal_ecrit_2 = %d\n",mal_ecrit_2(y,x)); // utilise nb = 3,
     renvoie nb <- 0
    // Attention, on a appele la fonction mal_ecrit_2 avec l'ordre de y et x
    inverse
    printf ("nb = %d\n",nb);
    return 0;
}
```

```
29: nb main = 3
30: nb mal_ecrit_1 = 3
    nb mal_ecrit_1 = 14
    nb mal_ecrit_1 = 4
31: nb main = 3
32: nb mal_ecrit_2 = 9
    nb mal_ecrit_2 = 6
    resultat de mal_ecrit_2 = 0
34: nb = 0
```

Travaux Dirigés n°5

Travaux Dirigés n°6

Exercice 1. *Calculer π en appliquant la suite de Leibniz :* $\pi/4 = 1-1/3+1/5-1/7+1/9-\ldots$.

Une boucle `while` me semble la plus appropriée parce que on ne sait pas exactement le nombre de fois que l'on devrait itérer l'alorithme.

Algorithm 1 Crible d'Érastosthène

Données d'entrée : T (tableau d'entiers > 0), N (nombre entier)
Données de sortie : T avec les éléments non-primes remplacés par -1

1: $T \leftarrow 2 : N$
2: **for** $i \leftarrow 2, \left\lfloor \sqrt{N} \right\rfloor$ **do**
3: **for** $j \leftarrow 1, N$ **do**
4: **if** $T_j \bmod i \equiv 0$ **then**
5: $T_j \leftarrow -1$
6: **end if**
7: **end for**
8: **end for**
9: **return** T