

Synthèse TD4

Actions et structure de contrôle

Introduction

- Les objectifs : définir un algorithme à partir d'un énoncé donc
 - -l'environnement
 - -le lexique des variables
 - -la séquence algorithmique
 - -le jeu d'essai

Exercice 1

Énoncé :

Une station climatique du Sénégal fournit un certain nombre de couples de températures relevées chaque jour, la première étant le minimum observé, la deuxième le maximum. Certain couple possède une et une seule donnée manquante, notée 0, qui n'est pas une valeur possible au Sénégal. Le dernier couple est suivi par $(0,0)$.

Exercice 1

Travail :

Déterminez un algorithme permettant de calculer les moyennes des minima et maxima, le nombre de lacunes (valeur manquante) et le pourcentage qu'elles représentent.

Exercice 1

Le début :

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5
6
7  {
8      float valMin;
9      float valMax;
10     float sommeMax;
11     float sommeMin;
12     float cptMax;
13     float cptMin;
14     float cpt0;
15
16
17     sommeMax=0;
18     sommeMin=0;
19     cptMax=0;
20     cptMin=0;
21     cpt0=0;
```

```
22     cpt0=0;
23     cptMin=0;
24     cptMax=0;
25     sommeMin=0;
26     sommeMax=0;
```

Exercice 1

On commence par faire le lexique :

```
7  □ {  
8      float valMin;  
9      float valMax;  
10     float sommeMax;  
11     float sommeMin;  
12     float cptMax;  
13     float cptMin;  
14     float cpt0;  
15  
16     float cpt0;  
17     float cptMin;  
18     float cptMax;
```

Exercice 1

Puis on initialise les variables

16	
17	sommeMax=0;
18	sommeMin=0;
19	cptMax=0;
20	cptMin=0;
21	cpt0=0;
22	

Exercice 1

Le codage :

```
26 do
27 {
28     printf("temperature minimum : \n");
29     scanf("%f", &valMin);
30
31
32     if (valMin==0)
33     {
34         cpt0=cpt0+1;
35     }
36     else
37     {
38         cptMin=cptMin+1;
39         sommeMin=sommeMin+valMin;
40     }
41
42     printf("temperature maximal : \n");
43     scanf("%f", &valMax);
44     if (valMax==0)
45     {
46         cpt0=cpt0+1;
47     }
48     else
49     {
50         cptMax=cptMax+1;
51         sommeMax=sommeMax+valMax;
52     }
53 }
54 while (valMax!=0 || valMin!=0);
55
56 sommeMax=sommeMax/cptMax;
57 sommeMin=sommeMin/cptMin;
58 cpt0=cpt0-2;
59 cpt0=cpt0/(cptMax+cptMin+cpt0)*100;
60
61 printf("moyenneMax : %.3f\n", sommeMax);
62 printf("moyenneMin : %.3f\n", sommeMin);
63 printf("valeur manquante : %.2f pourcent\n", cpt0);
64
65 return 0;
66 }
67
68 return 0;
```


Exercice 1

Faire une boucle FAIRE... TANTQUE :

```
26  do
27  {
28      printf("temperature minimun : \n");
29      scanf("%f", &valMin);
30
31
32      if (valMin==0)
33      {
34          cpt0=cpt0+1;
35      }
36      else
37      {
38          cptMin=cptMin+1;
39          sommeMin=sommeMin+valMin;
40      }
41
42      printf("temperature maximal : \n");
43      scanf("%f", &valMax);
44      if (valMax==0)
45      {
46          cpt0=cpt0+1;
47      }
48      else
49      {
50          cptMax=cptMax+1;
51          sommeMax=sommeMax+valMax;
52      }
53  }
54  while (valMax!=0 || valMin!=0);
```

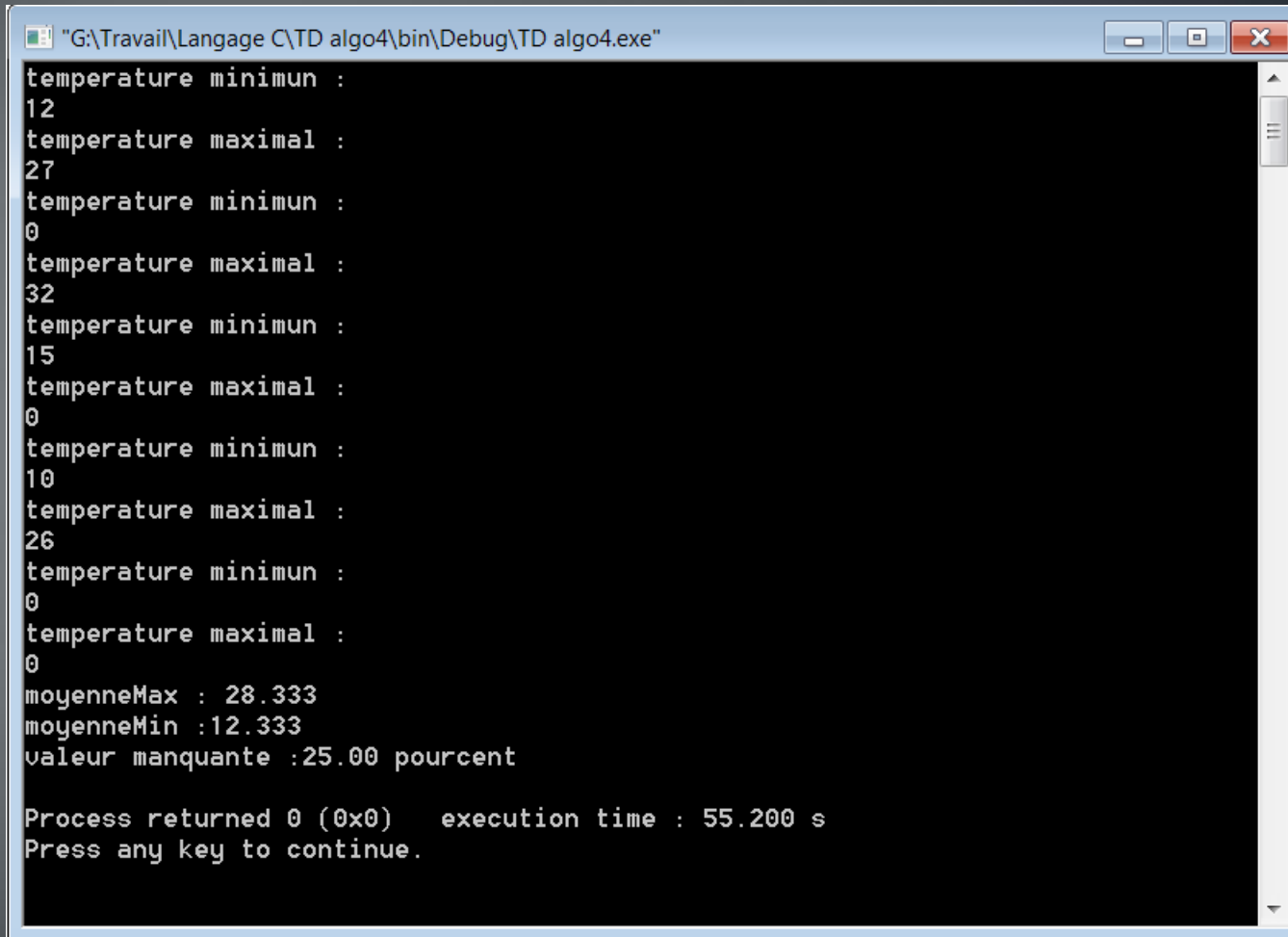
Exercice 1

Écrire les résultats :

```
56     sommeMax=sommeMax/cptMax;
57     sommeMin=sommeMin/cptMin;
58     cpt0=cpt0-2;
59     cpt0=cpt0/(cptMax+cptMin+cpt0)*100;
60
61     printf("moyenneMax : %.3f\n",sommeMax);
62     printf("moyenneMin :%.3f\n",sommeMin);
63     printf("valeur manquante :%.2f pourcent\n",cpt0);
64
65     return 0;
66 }
```

```
ee     }
e2     return 0;
e4
e6     printf("valeur manquante :%.2f pourcent\n",cpt0);
```

Exercice 1



```
"G:\Travail\Langage C\TD algo4\bin\Debug\TD algo4.exe"
temperature minimum :
12
temperature maximal :
27
temperature minimum :
0
temperature maximal :
32
temperature minimum :
15
temperature maximal :
0
temperature minimum :
10
temperature maximal :
26
temperature minimum :
0
temperature maximal :
0
moyenneMax : 28.333
moyenneMin : 12.333
valeur manquante : 25.00 pourcent

Process returned 0 (0x0)   execution time : 55.200 s
Press any key to continue.
```

```
Press any key to continue.
Process returned 0 (0x0)   execution time : 22.500 s
```

Exercice 2

Enoncé :

Deux pays (P1 et P2) possèdent une population taillePopulation1 et taillePopulation2 et un taux de croissance annuel tauxPopulation1 et tauxPopulation2 .
Pour ces pays, les hypothèses suivantes s'appliquent :

$$\text{taillePopulation1} > \text{taillePopulation2}$$

$$\text{tauxPopulation1} < \text{tauxPopulation2}$$

Question :

Calculez au bout de combien d'années la population de P2 aura dépassé la population de P1.

Exercice 2

Lexique & Initialisation :

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      //lexique
7
8      float taillepopulation1;
9      float taillepopulation2;
10     float tauxpopulation1;
11     float tauxpopulation2;
12     int annee;
13
14     //initialisation
15
16     annee = 0;
17
18     //initialisation
19
20     //initialisation
21
22     //initialisation
23
24     //initialisation
25
```

Exercice 2

Début du programme :

```
17 //début
18
19
20 printf("la taille de la population 1 doit etre superieur a celui de la population 2.\n");
21 printf("le taux de la population 1 doit etre inferieur a celui de la population 2.\n");
22
23 printf("Entrer la taille de la population 1 :\n");
24 scanf("%f",&taillepopulation1);
25
26 printf("Entrer la taux de la population 1 en pourcentage:\n");
27 scanf("%f",&tauxpopulation1);
28
29 printf("Entrer la taille de la population 2 :\n");
30 scanf("%f",&taillepopulation2);
31
32 printf("Entrer la taux de la population 2 en pourcentage:\n");
33 scanf("%f",&tauxpopulation2);
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
```

Exercice 2

```
32 //action:
33
34
35 if(taillepopulation1<taillepopulation2 || tauxpopulation1>tauxpopulation2)
36 {
37     printf("ERREUR: la taille de la population 2 est superieur a celle de la population 1 ou le taux de population 1 est superieur à la 2 ");
38 }
39 else{do
40 {
41     taillepopulation1 = taillepopulation1 + (taillepopulation1 * (tauxpopulation1 / 100));
42     taillepopulation2 = taillepopulation2 + (taillepopulation2 * (tauxpopulation2 / 100));
43     annee = annee + 1;
44 }
45 while (taillepopulation2<taillepopulation1);
46
47 //fin action
48
49 //fin fonction
50
51
52 printf ("%d\n",taillepopulation1);
53 }
54
55 annee = annee + 1;
```

Exercice 2

```
49     printf("Au bout de %d ",annee);
50     printf("annee la population 2 aura depasse la population 1.\n");
51     }
52     printf("FIN DU PROGRAMME.");
53
54
55     //fin
56     return 0;
57 }
58
```

```
28 }
29
29     return 0;
30
```


Exercice 2

la taille de la population 1 doit etre superieur a celui de la population 2.
la taux de la population 1 doit etre inferieur a celui de la population 2.
Entrer la taille de la population 1 :

Exercice 2

```
la taille de la population 1 doit etre superieur a celui de la population 2.  
le taux de la population 1 doit etre inferieur a celui de la population 2.  
Entrer la taille de la population 1 :  
1000000  
Entrer la taux de la population 1 en pourcentage:  
1500000  
Entrer la taille de la population 2 :  
12  
Entrer la taux de la population 2 en pourcentage:  
5  
ERREUR: la taille de la population 2 est superieur a celle de la population 1 ou  
le taux de population 1 est superieur á la 2 FIN DU PROGRAMME.  
Process returned 0 (0x0)   execution time : 12.719 s  
Press any key to continue.
```

Exercice 2

```
la taille de la population 1 doit etre superieur a celui de la population 2.  
la taux de la population 1 doit etre inferieur a celui de la population 2.  
Entrer la taille de la population 1 :  
150000  
Entrer la taux de la population 1 en pourcentage:  
5  
Entrer la taille de la population 2 :  
100000  
Entrer la taux de la population 2 en pourcentage:  
50  
Au bout de 2 annee la population 2 aura depasse la population 1.  
FIN DU PROGRAMME.  
Process returned 0 (0x0)   execution time : 12.156 s  
Press any key to continue.  
-
```

Exercice 3

Enoncé:

Créer un algorithme permettant de calculer π à 10^{-4} grâce à la somme donnée qui tend vers $\pi/4$:

$$1 - 1/3 + 1/5 - 1/7 + 1/9 - 1/11 + 1/13 \dots$$

Exercice 3

A quoi sert cette somme?

Tout d'abord on sait que $\pi/4$ est égale à environ
0,7853981634.

Exercice 3

1	- 1/3	0,66667		0,76979	1/33	0,80009
0,66667	1/5	0,86667		0,80009	- 1/35	0,77152
0,86667	- 1/7	0,72381		0,77152	1/37	0,79855
0,72381	1/9	0,83492		0,79855	- 1/39	0,77291
0,83492	- 1/11	0,74401		0,77291	1/41	0,79730
0,74401	1/13	0,82093		0,79730	- 1/43	0,77404
0,82093	- 1/15	0,75427		0,77404	1/45	0,79626
0,75427	1/17	0,81309		0,79626	- 1/47	0,77499
0,81309	- 1/19	0,76046		0,77499	1/49	0,79539
0,76046	1/21	0,80808		0,79539	- 1/51	0,77579
0,80808	- 1/23	0,76460		0,77579	1/53	0,79465
0,76460	1/25	0,80460		0,79465	- 1/55	0,77647
0,80460	- 1/27	0,76756		0,77647	1/57	0,79402
0,76756	1/29	0,80205		0,79402	- 1/59	0,77707
0,80205	- 1/31	0,76979		0,77707	1/61	0,79346

Exercice 3

Ensuite il faut trouver le style de boucle que nous allons utiliser:

Sachant que l'on doit faire une boucle jusqu'à trouver un encadrement plus précis pour $\pi/4$ il faut faire une boucle faire tant que avec une condition que nous allons déterminer

Exercice 3

```
22
23 do
24 {
25     totalancien = total;
26     total = total - (1 / (float)diviseur) * multi;
27     diviseur = diviseur + 2;
28     multi = multi * - 1;
29 }
30 while ( totalancien - total >= precision || total - totalancien >= precision);
31
32 total = total * 4;
33
34 total = total * 4;
35
36 while ( totalancien - total >= precision || total - totalancien >= precision);
```


Exercice 3

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      //lexique:
7
8      int diviseur;
9      float total;
10     float totalancien;
11     float multi;
12     float precision;
13
14
15     //initialisation:
16
17     total = 1;
18     diviseur = 3;
19     multi = 1;
20     totalancien = 0;
21     precision = 0.0001 / 4; //pour pi/4
22
23
24     //calcul de pi/4:
25     while (precision > 0.0001)
26     {
27         multi = 1;
28         diviseur = 3;
29         total = 1;
30         totalancien = 0;
31         while (total - totalancien > precision)
32         {
33             totalancien = total;
34             multi = multi * 1.0 / diviseur;
35             total = total + multi;
36             diviseur = diviseur * 2;
37         }
38         precision = precision * 2;
39     }
40     printf("pi/4 = %f\n", total);
41     return 0;
42 }
```

Exercice 3

```
34      //affichage:
35
36      printf("pi est egale a ");
37      printf("%.4f",total);
38
39
40      return 0;
41  }
42
43
44  }
45      return 0;
```

Exercice 4

Nous pouvons représenter un sablier comme ci-dessous avec une suite de nombres dont le plus grand sur la première ligne doit être impaire. Par l'exemple, ici premierN vaut 5.

1	2	3	4	5
0	6	7	8	0
0	0	9	0	0
0	10	11	12	0
13	14	15	16	17

Question :

Écrivez un algorithme qui permet de réaliser la figure précédente représentant un sablier. Vous le testerez avec 5, puis avec des nombres de votre choix.

Conclusion

Les exercices nous ont familiarisé avec le langage C notamment pour les boucles.

Les principales difficultés des exercices étaient la compréhension des sujets, en particuliers le 3 avec Pi qui était plutôt complexe avec les conditions de sortie.