

## Travaux dirigés 9 : fonctions et procédures (2)

### 1 Trace de fonctions

1. Faire la trace du programme suivant.

```
1  /* Declaration de fonctionnalites supplementaires */
2  #include <stdlib.h> /* EXIT_SUCCESS */
3  #include <stdio.h> /* printf() */
4
5  /* Declarations constantes et types utilisateurs */
6
7  /* Declarations de fonctions utilisateurs */
8  void permute_valeurs(int a,int b);
9
10 /* Fonction principale */
11 int main()
12 {
13     /* Declaration et initialisation des variables */
14     int x = 1;
15     int y = 2;
16
17     permute_valeurs(x,y);
18     printf("x = %d et y = %d\n",x,y);
19
20     /* Valeur fonction */
21     return EXIT_SUCCESS;
22 }
23
24 /* Definitions de fonctions utilisateurs */
25 void permute_valeurs(int a,int b)
26 {
27     int aux; /*var aux pour permutation */
28
29     aux = a;
30     a = b;
31     b = aux;
32 }
```

**Correction.** L'échange des valeurs de **a** et **b** a lieu, par contre l'échange des valeurs de **x** et **y** n'a pas lieu, puisque `permute_valeurs` n'accède pas aux cases mémoires du `main` seulement à ses propres cases mémoires.

### 2 Le menu avec fonctions et procédures

Dans cet exercice vous complétez le menu écrit en TP. Vous travaillerez sur trois parties du programme :

- les déclarations du début du programme (fonctionnalités, constantes, fonctions),
- les définitions de fonctions,
- la fonction principale (`main`).

```
***** MENU *****
*
* 1) Tester si un nombre est premier *
* 2) Calcul de factorielle           *
* 3) Deviner un nombre               *
* 4) Motif d'etoiles                 *
*
* 0) QUITTER                         *
*
***** votre choix :

```

2. Déclarer et définir une procédure qui affichera le menu.

main()

ligne	x	y	Affichage (sortie écran)	
initialisation	1	2		
17				

permute_valeurs(1, 2)				
ligne	a	b	aux	Affichage
initialisation	1	2	?	
29			1	
30	2			
31		1		
32	ne renvoie rien			

18			x = 1 et y = 2
21	SORTIE AVEC SUCCÈS		

TABLE 1 – Trace du programme de l'exercice 1.

**Correction.** C'est tout bête.

```
/* Declarations de fonctions utilisateurs */
void afficher_menu();
...
/* Definitions de fonctions utilisateurs */
void afficher_menu()
{
    printf("***** MENU *****\n");
    printf("*\n");
    printf("* 1) Tester si un nombre est premier *\n");
    printf("* 2) Calcul de factorielle *\n");
    printf("* 3) Deviner un nombre *\n");
    printf("* 4) Motif d'etoiles *\n");
    printf("*\n");
    printf("* 0) QUITTER *\n");
    printf("*\n");
    printf("***** votre choix :");
}
```

3. Déclarer et définir une fonction `choix_utilisateur`, sans paramètres qui renverra une valeur entière saisie par l'utilisateur.

**Correction.**

```
int choix_utilisateur()
{
    int choix;
    scanf("%d", &choix);
    return choix;
}
```

4. Déclarer et définir une fonction `executer_menu` qui :
  - affichera le menu à l'utilisateur et réalisera la saisie de son choix ;
  - lorsque ce choix est 1, appellera une fonction non encore définie `menu_premier` ;
  - puis, lorsque ce choix est différent de 0, renverra `TRUE` et lorsque ce choix est égal à zéro renverra `FALSE`.

**Correction.** Les étudiants doivent se poser la question de ce dont pourrait avoir besoin `menu_premier` pour effectuer le traitement et ce qu'elle doit renvoyer (réponse : rien dans les deux cas).

```
int executer_menu()
{
    int choix;

    /* Affichage du menu et choix de l'utilisateur */
    afficher_menu();
```

```

    choix = choix_utilisateur();

    if (1 == choix) /* ----- 1) Tester si un nombre est premier ----- */
    {
        menu_premier();
    }

    /* Valeur de retour */
    if (choix != 0)
    {
        return TRUE;
    }
    return FALSE;
}

```

5. Déclarer et définir une procédure `menu_premier` qui traitera le choix 1, et qui fera appel à la fonction `est_premier` (dont vous rappellerez la définition et la déclaration) et à la fonction `choix_utilisateur` pour le choix de l'entier.

### Correction.

```

void menu_premier()
{
    int p;

    printf("Donner un nombre : ");
    p = choix_utilisateur();

    if (est_premier(p))
    {
        printf("Le nombre %d est premier\n", p);
    }
    else
    {
        printf("Le nombre %d n'est pas premier\n", p);
    }
}

```

```

Rappel :
int est_premier(int n)
{
    int i = 2;
    int premier = TRUE;

    while (premier && i < n)
    {
        if (n % i == 0)
        {
            premier = FALSE;
        }
        i = i + 1;
    }
    return premier;
}

```

6. Écrire le `main` de telle sorte qu'il fasse appel à la fonction `executer_menu` tant que celle-ci renvoie `TRUE`.

### Correction.

```

int main()
{
    /* Declarations et initialisation des variables */
    int encore = TRUE;

    /* Boucle d'interaction avec l'utilisateur */
    while (encore)
    {
        encore = executer_menu();
    }
}

```

```

    }

    /* Greetings */
    printf("Bye bye\n");

    /* Valeur fonction */
    return EXIT_SUCCESS;
}

```

7. Faire la trace de votre programme dans le cas où l'utilisateur saisit 1 puis 3 puis 0.

**Correction.** Avant de faire la trace il nous faut des numéros de ligne :

```

1  /* Fonctionnalités supplémentaires */
2  #include <stdlib.h> /* EXIT_SUCCESS */
3  #include <stdio.h> /* printf(), scanf() */
4
5  /* Déclarations de types et constantes utilisateurs */
6  #define TRUE 1
7  #define FALSE 0
8
9  /* Déclarations de fonctions utilisateurs */
10
11 /* Affiche le menu */
12 void afficher_menu();
13
14 /* Récupère un nombre entier saisi par l'utilisateur et le retourne */
15 int choix_utilisateur();
16
17 /* Affichage du menu et traitement du choix de l'utilisateur */
18 int executer_menu();
19
20 /* Teste si un nombre saisi par l'utilisateur est premier et affiche le résultat */
21 void menu_premier();
22
23 /* Teste si son argument (positif) est premier */
24 int est_premier(int p);
25
26
27 /* Fonction principale */
28 int main()
29 {
30     /* Déclarations et initialisation des variables */
31     int encore = TRUE;
32
33     /* Boucle d'interaction avec l'utilisateur */
34     while (encore)
35     {
36         encore = executer_menu();
37     }
38
39     /* Greetings */
40     printf("Bye bye\n");
41
42     /* Valeur fonction */
43     return EXIT_SUCCESS;
44 }
45
46
47
48 /* Définitions de fonctions utilisateurs */
49 void afficher_menu()
50 {
51     printf("***** MENU *****\n");
52     printf("*\n");
53     printf("* 1) Tester si un nombre est premier *\n");
54     printf("* 2) Calculatrice *\n");
55     printf("* 3) Deviner un nombre *\n");
56     printf("* 4) Motif d'étoiles *\n");
57     printf("*\n");

```

```

58     printf("* 0) QUITTER                *\n");
59     printf("*\n");
60     printf("***** votre choix :");
61 }
62
63 int choix_utilisateur()
64 {
65     int choix;
66     scanf("%d", &choix);
67     return choix;
68 }
69
70 int executer_menu()
71 {
72     int choix;
73
74     /* Affichage du menu et choix de l'utilisateur */
75     afficher_menu();
76     choix = choix_utilisateur();
77
78     if (1 == choix) /* ----- 1) Tester si un nombre est premier ----- */
79     {
80         menu_premier();
81     }
82
83     /* Valeur de retour */
84     if (choix != 0)
85     {
86         return TRUE;
87     }
88     return FALSE;
89 }
90
91 void menu_premier()
92 {
93     int p;
94
95     printf("Donner un nombre : ");
96     p = choix_utilisateur();
97
98     if (est_premier(p))
99     {
100         printf("Le nombre %d est premier\n", p);
101     }
102     else
103     {
104         printf("Le nombre %d n'est pas premier\n", p);
105     }
106 }
107
108
109 int est_premier(int n)
110 {
111     int i = 2;
112     int premier = TRUE;
113
114     while (premier && i < n)
115     {
116         if (n % i == 0)
117         {
118             premier = FALSE;
119         }
120         i = i + 1;
121     }
122     return premier;
123 }

```

On peut maintenant faire la trace (Table 2 page 13). La trace est donnée en entier, mais inutile de tout faire. Disons qu'à partir du second appel à `executer_menu` c'est un peu

superflu.

8. Modifier votre fonction `choix_utilisateur` de telle sorte que :
  - elle prenne en argument deux paramètres entiers  $a$  et  $b$  ;
  - si l'utilisateur saisit un nombre  $n \in [a, b]$  la fonction retourne  $n$  sans générer d'affichage ;
  - si l'utilisateur saisit un nombre  $n \notin [a, b]$  l'intervalle de saisie soit affiché à l'utilisateur et la saisie redemandée, jusqu'à cinq fois.

**Correction.** Au delà de cinq essais, c'est la dernière saisie utilisateur qui est renvoyée.

```
int choix_utilisateur(int a, int b)
{
    int compteur = 5; /* compteur du nombre d'essais */
    int choix; /* choix de l'utilisateur */

    scanf("%d", &choix);
    while ((compteur > 0) && ((choix < a) || (choix > b)))
    {
        printf("Le nombre doit etre entre %d et %d (inclus) : ", a, b);
        scanf("%d", &choix);
        compteur = compteur - 1;
    }
    return choix;
}
```

IL faut aussi modifier les appels à cette fonction. Pour la fonction `executer_menu` on donne l'intervalle  $[0, 3]$  (à supposer que ces quatres choix soient traités). Pour la procédure `menu_premier`, on donne l'intervalle  $[1, \text{INT\_MAX}]$  (charger `limits.h` pour `INT\_MAX`).

On se donne la procédure suivante :

```
void afficher_motif(int cote)
{
    int ligne; /* numero de ligne, de bas en haut */
    int colonne; /* numero de colonne, de gauche a droite */
    for (ligne = cote - 1; ligne >= 0; ligne = ligne - 1) /* pour chaque ligne */
    {
        for (colonne = 0; colonne < cote; colonne = colonne + 1) /* pour chaque colonne */
        {
            if (motif(colonne, ligne)) /* le point appartient au motif */
            {
                printf("* ");
            }
            else /* le point n'appartient pas au motif */
            {
                printf("  ");
            }
        }
        printf("\n"); /* ligne suivante */
    }
}
```

9. Définir la fonction motif de telle sorte qu'un appel à `afficher_motif(5)` affiche :

```

*           *
*           *
*       *
*   *
* * * * *

```

**Correction.** Facile :

```
int motif(int x, int y)
{
    return x == y
        || x == 0
        || y == 0;
}
```

10. Écrire les fonctions nécessaires au traitement du choix 2 et du choix 3 du menu sur le modèle du traitement du choix 1.

**Correction.** Il faut écrire une procédure `menu_deviner`, qui fera appel à une fonction `tirage_aleatoire` puis en boucle à `choix_utilisateur` sur l'intervalle dans lequel le nombre est tiré (de 0 à une constante symbolique).

Le code complet du programme à la fin du TD (mais personne ne va finir non ?) :

```
1  /* Fonctionnalites supplementaires */
2  #include <stdlib.h> /* EXIT_SUCCESS, rand(), srand() */
3  #include <stdio.h> /* printf(), scanf() */
4  #include <limits.h> /* INT_MAX */
5  #include <time.h> /* time() */
6
7  /* Declarations de types et constantes utilisateurs */
8  #define TRUE 1
9  #define FALSE 0
10 #define DEVINER_MAX 100
11 #define DEVINER_ESSAIS 8
12
13 /* Declarations de fonctions utilisateurs */
14
15 /* Affiche le menu */
16 void afficher_menu();
17
18 /* Recupere un nombre entier saisi par l'utilisateur dans l'intervalle
19  * [a,b] et le retourne */
20 int choix_utilisateur(int a, int b);
21
22 /* Affichage du menu et traitement du choix de l'utilisateur */
23 int executer_menu();
24
25 /* Teste si un nombre saisi par l'utilisateur est premier et affiche le resultat */
26 void menu_premier();
27
28 /* Teste si son argument (positif) est premier */
29 int est_premier(int p);
30
31 /* Factorielle */
32 void menu_factorielle();
33 int factorielle(int n);
34
35 /* Jouer a une devinette */
36 void menu_deviner();
37
38 /* Tirer un nombre au hasard entre 0 et n */
39 int nombre_aleatoire(int n);
40
41 /* calculette */
42 void menu_calculatrice();
43 int calculatrice();
44
45 /* motif */
46 void afficher_motif(int cote);
47 int motif(int colonne, int ligne);
48
49 /* Fonction principale */
50 int main()
51 {
52     /* Declarations et initialisation des variables */
53     int encore = TRUE;
54
55     /* Initialisation du generateur aleatoire */
56     srand(time(NULL)); /* à ne faire qu'une fois */
57
58     /* Boucle d'interaction avec l'utilisateur */
59     while (encore)
60     {
```

```

61         encore = executer_menu();
62     }
63
64     /* Greetings */
65     printf("Bye bye\n");
66
67     /* Valeur fonction */
68     return EXIT_SUCCESS;
69 }
70
71
72
73 /* Definitions de fonctions utilisateurs */
74 void afficher_menu()
75 {
76     printf("***** MENU *****\n");
77     printf("*\n");
78     printf("* 1) Tester si un nombre est premier *\n");
79     printf("* 2) Calcul de factorielle *\n");
80     printf("* 3) Deviner un nombre *\n");
81     printf("* 4) Motif d'etoiles *\n");
82     printf("* 5) Calculette *\n");
83     printf("*\n");
84     printf("* 0) QUITTER *\n");
85     printf("*\n");
86     printf("***** votre choix : ");
87 }
88
89 int choix_utilisateur(int a, int b)
90 {
91     int compteur = 5; /* compteur du nombre d'essais */
92     int choix = 0; /* choix de l'utilisateur */
93
94     scanf("%d", &choix);
95     while ((compteur > 0) && ((choix < a) || (choix > b)))
96     {
97         printf("Le nombre doit etre entre %d et %d (inclus) : ", a, b);
98         scanf("%d", &choix);
99         compteur = compteur - 1;
100     }
101     return choix;
102 }
103
104 int executer_menu()
105 {
106     int choix;
107
108     /* Affichage du menu et choix de l'utilisateur */
109     afficher_menu();
110     choix = choix_utilisateur(0, 5);
111
112     if (1 == choix) /* ----- 1) Tester si un nombre est premier ----- */
113     {
114         menu_premier();
115     }
116
117     if (2 == choix) /* ----- 2) calcul de factorielle ----- */
118     {
119         menu_factorielle();
120     }
121
122     if (3 == choix) /* ----- 3) Deviner un nombre ----- */
123     {
124         menu_deviner();
125     }
126
127     if (4 == choix) /* ----- 4) Motif d'etoiles ----- */
128     {
129         afficher_motif(10);

```



```

130     }
131
132     if (5 == choix) /* ----- 5) Calculatrice ----- */
133     {
134         menu_calculatrice();
135     }
136
137     /* Valeur de retour */
138     if (choix != 0)
139     {
140         return TRUE;
141     }
142     return FALSE;
143 }
144
145 void menu_premier()
146 {
147     int p;
148
149     printf("Donner un nombre : ");
150     p = choix_utilisateur(1, INT_MAX);
151
152     if (est_premier(p))
153     {
154         printf("Le nombre %d est premier\n", p);
155     }
156     else
157     {
158         printf("Le nombre %d n'est pas premier\n", p);
159     }
160 }
161
162
163 int est_premier(int n)
164 {
165     int i = 2;
166     int premier = TRUE;
167
168     while (premier && i < n)
169     {
170         if (n % i == 0)
171         {
172             premier = FALSE;
173         }
174         i = i + 1;
175     }
176     return premier;
177 }
178
179 void menu_factorielle()
180 {
181     int n;
182     printf("Calcul de factorielle.\nEnter l'argument (entier positif) : ");
183     n = choix_utilisateur(0, INT_MAX);
184     printf("%d\n", factorielle(n));
185 }
186
187 int factorielle(int n)
188 {
189     int i;
190     int res = 1;
191     for (i = 2; i <= n; i = i + 1)
192     {
193         res = res * i;
194     }
195     return res;
196 }
197
198 void menu_deviner()

```

```

199 {
200     int choix; /* choix de l'utilisateur pour le nombre secret */
201     int trouve = FALSE; /* TRUE si trouvé */
202     int nombre_secret;
203     int essais = DEVINER_ESSAIS; /* essais restants */
204
205     /* Tirage aléatoire du nombre secret */
206     nombre_secret = nombre_aleatoire(DEVINER_MAX);
207
208     /* manche joueur */
209     while(!trouve && (essais > 0)) /* pas trouvé nombre secret */
210     {
211         /* demande nombre à l'utilisateur */
212         printf("Votre choix (nombre entre 0 et %d) : ", DEVINER_MAX);
213         choix = choix_utilisateur(0, DEVINER_MAX);
214
215         if(choix == nombre_secret) /* trouvé */
216         {
217             trouve = TRUE;
218         }
219         else /* pas trouvé */
220         {
221             /* donne indice */
222             if(choix > nombre_secret)
223             {
224                 printf("Trop grand.\n");
225             }
226             else
227             {
228                 printf("Trop petit.\n");
229             }
230         }
231         essais = essais - 1;
232     }
233
234     if (essais > 0)
235     {
236         /* trouvé nombre secret */
237         printf("Gagné. Vous avez trouvé le nombre secret.\n");
238     }
239     else
240     {
241         /* Perdu */
242         printf("Perdu : limite du nombre d'essais atteinte.\n");
243     }
244 }
245
246
247 int nombre_aleatoire(int n)
248 {
249     /* tirage du nombre secret */
250     return rand() % (n + 1); /* entre 0 et n inclus */
251 }
252
253 void menu_calcullette()
254 {
255     int continuer = TRUE;
256     printf("Mode calcullette\n");
257     printf("Entrer des expressions nombre operateur nombre\n");
258     printf("ou 0+0 pour quitter\n");
259     while (continuer)
260     {
261         printf("Votre expression ? ");
262         continuer = calcullette();
263     }
264 }
265
266 int calcullette()
267 {

```

```

268     double x;
269     double y;
270     char op;
271
272     scanf("%lg %c %lg", &x, &op, &y);
273
274     if ('+' == op) /* addition */
275     {
276         if ((0 == x) && (0 == y))
277         {
278             printf("La tete a toto \n");
279             return FALSE;
280         }
281         printf("%lg\n", x + y);
282     }
283     else if ('-' == op) /* soustraction */
284     {
285         printf("%lg\n", y - x);
286     }
287     else if (('*' == op) || ('x' == op)) /* multiplication */
288     {
289         printf("%lg\n", x * y);
290     }
291     else if ( '/' == op) /* division */
292     {
293         printf("%lg\n", x / y);
294     }
295     else
296     {
297         printf("operation non reconnue\n");
298         return FALSE;
299     }
300     return TRUE;
301 }
302
303 void afficher_motif(int cote)
304 {
305     int ligne; /* numero de ligne, de bas en haut */
306     int colonne; /* numero de colonne, de gauche a droite */
307     for (ligne = cote - 1; ligne >= 0; ligne = ligne - 1) /* pour chaque ligne */
308     {
309         for (colonne = 0; colonne < cote; colonne = colonne + 1) /* pour chaque colonne */
310         {
311             if (motif(colonne, ligne)) /* le point appartient au motif */
312             {
313                 printf("* ");
314             }
315             else /* le point n'appartient pas au motif */
316             {
317                 printf(" ");
318             }
319         }
320         printf("\n"); /* ligne suivante */
321     }
322 }
323
324 int motif(int colonne, int ligne)
325 {
326     return (0 == colonne)
327         || (0 == ligne)
328         || (ligne == colonne);
329 }

```

### 3 En travaux pratiques

Le programme de ce TD est à reprendre en TP. Votre code doit être correctement indenté, sinon il sera difficile d'y trouver vos erreurs. Vous pouvez utiliser la commande `astyle menu.c`

ou `~boudes/pub/bin/astyle menu.c` qui indentera correctement votre code. Alternative : l'éditeur de texte **emacs**, beaucoup plus puissant que **gedit**, gère parfaitement l'indentation.

Ajouter au menu une entrée **calculatrice** et une procédure **calculatrice** qui affichera le résultat d'une expression **nombre opération nombre** entrée par l'utilisateur, où les nombres sont des **double** et l'opération un caractère parmi +, -, \*, /. Indication :

```
...
scanf("%lg %c %lg", &x, &op, &y);
if ('+' == op)/* faire une addition */
{
    printf("%lg\n", x + y); /* affichage du résultat */
}
```

Des questions supplémentaires sont disponibles sur la page web du cours.

main()				
ligne	encore	Affichage		
initialisation	1 (vrai)			
36				
executer_menu()				
ligne	choix	Affichage		
init.	?			
75				
afficher_menu()				
ligne	Affichage			
initialisation				
51	*...			
:	:			
:	:			
60	...* votre choix :			
61	ne renvoie rien			
76				
choix_utilisateur() saisie 1				
ligne	choix	Affichage		
initialisation	?			
66	1			
67	renvoie 1			
76	1			
80				
menu_premier()				
ligne	p	Affichage		
initialisation	?			
96		...un nombre :		
97				
choix_utilisateur() saisie 1				
ligne	choix	Affichage		
initialisation	?			
66	3			
67	renvoie 3			
97	3			
98				
est_premier(3)				
ligne	n	i	premier	Affichage
init.	3	2	1 (vrai)	
120		3		
122	renvoie vrai (1)			
100			...3 est premier	
106	ne renvoie rien			
86	renvoie vrai (1)			
36	1 (vrai)			
36				
executer_menu()				
ligne	choix	Affichage		
initialisation	?			
75				
afficher_menu()				
ligne	Affichage			
initialisation				
51	*...			
:	:			
:	:			
60	...* votre choix :			
61	ne renvoie rien			
76				
choix_utilisateur() saisie 0				
ligne	choix	Affichage		
initialisation	?			
66	0			
67	renvoie 0			
76	0			
86	renvoie faux (0)			
36	0 (faux)			
40		Bye bye		
43	SUCCÈS			

TABLE 2 – Trace du programme de l'exercice 2.