Université Paris-Nord — L1
Institut Galilée — Année 2008-2009
Licence 1 - 1er semestre

Travaux dirigés 8 : Structures de contrôle “for”et “while”

Correction. Note aux chargés de TD : le début du TD (1h30, 2h) doit être consacré à la correction du partiel.

Résoudre les questions suivantes en utilisant soit *for*, soit *while*.

1 Nombres premiers

Diviseurs. Soient p et q deux entiers. On dit que q *divise* p, ou encore que q est un diviseur de p lorsqu'il existe un entier k tel que $p = k.q$.

Pour tester si q divise p, il suffit de calculer le reste de la division euclidienne de p par q. Le reste est nul si et seulement si q divise p. En C, ce reste est donné par l'expression `p % q`.

Nombres premiers. On dit d'un entier p qu'il est *premier* lorsqu'il n'est divisible que par 1 et par lui-même.

Questions :

- Donner un exemple de nombre premier.
- Écrire un programme qui demande à l'utilisateur un entier positif, teste si celui-ci est premier et affiche le résultat. Exemple d'exécution :

```
Donner un nombre entier positif : 89
Le nombre 89 est premier

Donner un nombre entier positif : 45
Le nombre 45 n'est pas premier, car 3 divise 45
```

Correction. L'algorithme utilisé est le suivant. On demande p à l'utilisateur. Pour chaque nombre entre 2 et $p-1$, on teste s'il divise p. On s'arrête dès qu'on trouve un diviseur, raison pour laquelle on utilise un while.

```
1  /* Test de primalité */
2  #include <stdlib.h> /* EXIT_SUCCESS */
3  #include <stdio.h> /* printf, scanf */
4
5  #define TRUE 1
6  #define FALSE 0
7
8  /* declaration de fonctions utilisateurs */
```

```
 9
10  int main()
11  {
12      int p; /* nombre a tester */
13      int q = 2; /* var. de boucle, diviseur de p */
14      int est_premier = TRUE;
15
16      /* Saisie utilisateur */
17      printf("Donner un nombre entier positif : ");
18      scanf("%d", &p);
19
20      /* Test de primalité */
21      while ( (q < p) && est_premier ) /* Pour  q < p, tant que q ne divise pas p */
22      {
23
24          if ( p % q == 0 ) /* q divise p */
25          {
26              est_premier = FALSE; /* p n'est pas premier */
27          }
28          q = q + 1; /* Passer au q suivant */
29      }
30
31      if (est_premier) /* p est premier ... ou négatif */
32      {
33          printf("Le nombre %d est premier\n", p);
34      }
35      else /* p n'est pas premier et q - 1 divise p */
36      {
37          printf("Le nombre %d n'est pas premier, car %d divise %d\n", p, q - 1, p);
38      }
39
40
41      /* valeur fonction */
42      return EXIT_SUCCESS;
43  }
44
45  /* implantation de fonctions utilisateurs */
```

2 Nombres parfaits

On dit d'un entier p qu'il est *parfait* lorsque la somme de ses diviseurs est égale à $2p$.

Questions

- Donner un exemple de nombre parfait.
- Écrire un programme qui demande à l'utilisateur un entier positif, teste si celui-ci est parfait et affiche le résultat. Exemple d'exécution :

```
Donner un nombre entier positif : 6
```

```
Le nombre 6 est parfait

Donner un nombre entier positif : 63
Le nombre 63 n'est pas parfait, la somme de ses diviseurs vaut 104
```

Correction. Cette fois ci l'algorithme impose un for car on parcourt tous les entiers entre 1 et n pour sommer ceux d'entre eux qui divisent n. Des optimisations sont bien entendu possibles, mais il vaut mieux éviter de les encourager.

```
1   /* Test de primalité */
2   #include <stdlib.h> /* EXIT_SUCCESS */
3   #include <stdio.h> /* printf, scanf */
4
5   #define TRUE 1
6   #define FALSE 0
7
8   /* declaration de fonctions utilisateurs */
9
10  int main()
11  {
12      int n; /* nombre a tester */
13      int i = 2; /* var. de boucle, diviseur de p */
14      int somme = 0;
15
16      /* Saisie utilisateur */
17      printf("Donner un nombre entier positif : ");
18      scanf("%d", &n);
19
20      /* Test de primalité */
21      for( i = 1; i < n;  i = i + 1) /* Pour i de 1 a p - 1 */
22      {
23          if ( n % i == 0 ) /* i est un diviseur de p */
24          {
25              somme = i + somme; /* ajoute i a la somme */
26          }
27      }
28
29      if (somme == n) /* n est parfait */
30      {
31          printf("Le nombre %d est parfait\n", n);
32      }
33      else /* p n'est pas premier et q - 1 divise p */
34      {
35          printf("Le nombre %d n'est pas parfait, la somme de ses diviseurs vaut %d\
36      }
37
38
39      /* valeur fonction */
```

```
40        return EXIT_SUCCESS;
41    }
42
43    /* implantation de fonctions utilisateurs */
```

3 En TP : qu'y a t-il au menu ?

Correction. Note aux chargés de TD.

- On incite ici les étudiants à structurer, indenter et commenter correctement leur code, sans quoi ils ne viendront pas à bout de ce type d'exercices.
- Cet exercice est également un exercice préparatoire à l'écriture et à l'appel de fonctions : nous referons l'exercice en utilisant des fonctions, les étudiants devraient normalement trouver ça plus claire.

Dans les TP précédents vous avez réalisé plusieurs programmes en C effectuant chacun une tâche. Le but de ce TP est de réunir plusieurs de ces programmes en un seul, dans lequel l'utilisateur choisira la tâche à effectuer dans un menu et où la fin d'une tâche débouchera sur un affichage du menu. Un exemple d'exécution est donné en fin de page.

3.1 Un menu

Dans un répertoire `TP8`, écrire et tester le programme `menu1.c` de manière à ce que :

- le programme affiche un menu proposant 3 choix représentés par des entiers : (1) tester si un entier est premier, (2) calculette ou (0) quitter. L'utilisateur fera son choix en entrant un entier.
- Si cet entier est 0, mettre fin au programme.
- Sinon : si cet entier est 1 afficher « premier : non disponible », si c'est 2, « calculette : non disponible » sinon « choix non disponible », puis réafficher le menu.

3.2 Des programmes

- Dans votre répertoire `TP8`, créer le programme `premier.c` qui teste si un nombre entré par l'utilisateur est premier (voir TD). Vérifier que votre programme fonctionne.
- Dans votre répertoire `TP6` (ou celui de votre binôme) doit se trouver le programme réalisant une mini-calculatrice. Vérifier que celui-ci est correctement indenté et commenté, et qu'il fonctionne.

3.3 Les programmes dans le menu

- Enregistrer le fichier `menu1.c` sous le nom `menu2.c`.
- Dans `premier.c` puis dans la calculatrice, copier le contenu du `main`, sauf le `return EXIT_SUCCESS;`, et coller le à la bonne place dans votre `menu2.c`. Remarque : pour copier/coller sous un système unix vous pouvez : (copier) sélectionner le texte à copier à l'aide de la souris ; (coller) effectuer un clic du milieu (bouton-molette) à l'endroit ou vous souhaitez coller.
- Tester `menu2.c` et s'il vous reste du temps ajouter des choix dans le menu.

```
****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Calculette                        *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 1
Donner un nombre entier positif : 34
Le nombre 34 n'est pas premier, 2 divise 34

****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Calculette                        *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 2
```

```
Entrer expression : nombre operateur nombre
4.001 * 0.7
4.001 * 0.7 = 2.8007


****************** MENU ******************
*                                        *
*   1) Tester si un nombre est premier   *
*   2) Calculette                        *
*   0) QUITTER                           *
*                                        *
************************* votre choix : 0


Sayonara
```

Correction. La correction est légérement différente de ce qui est demandé : avant de réafficher le menu, le programme demande à l'utilisateur de taper un caractère (différent d'un blanc).

```
1   #include <stdlib.h> /* EXIT_SUCCESS */
2   #include <stdio.h> /* printf, scanf */
3
4   #define TRUE 1
5   #define FALSE 0
6
7   /* declaration de fonctions utilisateurs */
8
9   int main()
10  {
11      int continuer = TRUE; /* TRUE si on doit proposer le menu */
12      int choix_menu = 0;   /* Choix de l'utilisateur */
13
14      /* Boucle principale d'interaction avec l'utilisateur */
15      while(continuer)
16      {
17
18          /* Affichage du menu */
19          printf("\n\n");
20          printf("****************** MENU ******************\n");
21          printf("*                                        *\n");
22          printf("*   1) Tester si un nombre est premier   *\n");
23          printf("*   2) Calculette                        *\n");
24          printf("*   ...                                  *\n");
25          printf("*                                        *\n");
26          printf("*                                        *\n");
27          printf("*   0) QUITTER                           *\n");
28          printf("*                                        *\n");
29          printf("************************* votre choix : ");
30
31          /* Choix utilisateur */
32          scanf("%d", &choix_menu);
```

```
33
34          /* Execution du choix de l'utilisateur (cas mutuellement exclusifs) */
35
36          if (1 == choix_menu) /* ---------------- Test de primalité -------------------- */
37          {
38              int p; /* nombre a tester */
39              int q = 2; /* var. de boucle, diviseur de p */
40              int est_premier = TRUE;
41
42              /* Saisie utilisateur */
43              printf("Donner un nombre entier positif : ");
44              scanf("%d", &p);
45
46              /* Test de primalité */
47              while ( (q < p) && est_premier ) /* Pour  q < p, tant que q ne divise pas p */
48              {
49
50                  if ( p % q == 0 ) /* i divise x */
51                  {
52                      est_premier = FALSE; /* p n'est pas premier */
53                  }
54                  q = q + 1; /* Passer au q suivant */
55              }
56
57              if (est_premier) /* p est premier ... ou négatif */
58              {
59                  printf("Le nombre %d est premier\n", p);
60              }
61              else /* x n'est pas premier et i - 1 divise x */
62              {
63                  printf("Le nombre %d n'est pas premier car %d divise %d\n", p, q - 1, p);
64              }
65          }
66
67          if (2 == choix_menu) /* --------------- Calculette -------------------------- */
68          {
69              double nombre_g; /* membre gauche de l'expression */
70              double nombre_d; /* membre droit de l'expression */
71              char op; /* operateur */
72              double expr; /* resultat de l'expression */
73
74              /* saisie expression */
75              printf("Entrez une expression de la forme : nombre operateur nombre\n");
76              scanf("%lg",&nombre_g);
77              scanf(" %c",&op);
78              scanf("%lg",&nombre_d);
79
80              /* calcul valeur expression */
81              /* cas mutuellement exclusif */
82              if(op == '+') /* addition */
83              {
```

```
84                  expr = nombre_g + nombre_d;
85              }
86
87              if(op == '-') /* soustraction */
88              {
89                  expr = nombre_g - nombre_d;
90              }
91
92              if(op == '*') /* multiplication */
93              {
94                  expr = nombre_g * nombre_d;
95              }
96
97              if(op == '/') /* division */
98              {
99                  expr = nombre_g / nombre_d;
100             }
101
102             /* affichage resultat */
103             printf("%g %c %g = %g\n",nombre_g,op,nombre_d,expr);
104         }
105
106         if (2 < choix_menu) /* Non disponible */
107         {
108             printf("\n** Choix non disponible **\n");
109         }
110
111
112         if (0 < choix_menu) /* Attendre que l'utilisateur soit pret a revenir au menu */
113         {
114             char c;
115             printf("\n[Saisir un caractere pour revenir au menu] ");
116             scanf(" %c", &c); /* un caractère blanc ne sera pas pris en compte */
117         }
118
119         if (0 == choix_menu) /* quitter */
120         {
121             printf("Sayonara\n");
122             continuer = FALSE;
123         }
124     }
125
126     /* valeur fonction */
127     return EXIT_SUCCESS;
128 }
129
130 /* implantation de fonctions utilisateurs */
131
```