

Carnet de Travaux Pratiques
Algorithmique et structures de données
Licence 2 Informatique

Julien BERNARD

Table des matières

Travaux Pratiques d'Algorithmique n°1	2
Exercice 1 : Mon premier programme en C	2
Exercice 2 : Découverte du langage C	2

Travaux Pratiques d'Algorithmique n°1

Exercice 1 : Mon premier programme en C

Le langage C est le langage que nous allons utiliser et apprendre pendant le cours d'Algorithmique. C'est un langage compilé, c'est-à-dire qu'il est nécessaire d'appeler un compilateur pour produire un exécutable qui sera fonctionnel uniquement sur la machine sur laquelle il a été compilé.

Question 1.1 Recopier le code source suivant dans un fichier appelé `hello.c`.

```
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```

Question 1.2 Compiler le programme avec la ligne de commande suivante :

```
gcc -Wall -std=c99 -O2 -o hello hello.c
```

Question 1.3 Exécuter le programme :

```
./hello
```

Exercice 2 : Découverte du langage C

Le but de cet exercice est de découvrir le langage C. Ce langage fait partie de la même famille que Java au niveau syntaxique, beaucoup de constructions sont similaires, de même que certains types.

Question 2.1 Recopier le code source suivant dans un fichier appelé `arg.c`.

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("Un argument est nécessaire !\n");
        return 1;
    }

    int arg = atoi(argv[1]);
    printf("L'argument est : %d\n", arg);

    return 0;
}
```

Dans ce programme, on utilise la ligne de commande pour fournir un nombre au programme. S'il n'y a pas assez d'argument, un message d'erreur est renvoyé et le programme s'arrête. Sinon, l'argument est transformé en nombre grâce à la fonction `atoi(3)`. Puis, on affiche le nombre grâce à `printf(3)`. Cette fonction prend comme argument une chaîne de caractères entre guillemets, puis éventuellement des noms d'identifiants de variables. Lors de l'exécution, la chaîne de caractères est affichée ainsi que la valeur de chaque variable à l'endroit indiqué. Les variables doivent apparaître dans l'ordre de leur apparitions dans la chaîne de caractères. Un entier est marqué grâce à `%d`, un flottant grâce à `%f`, un caractère grâce à `%c`.

```
$ ./arg 32
L'argument est : 32
```

On utilisera ce code comme base pour les questions suivantes.

Question 2.2 Faire un programme qui affiche la suite de Collatz. L'argument sera l'élément initial de la suite. Pour rappel, la suite de Collatz est définie par un élément initial u_0 strictement positif et :

$$u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n \text{ est divisible par 2} \\ 3u_n + 1 & \text{sinon} \end{cases}$$

On utilisera une boucle `while` et on s'arrêtera quand u_n vaudra 1.

```
$ ./collatz 46
46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1
```

Question 2.3 Faire un programme qui affiche les nombres de 1 à n où n est passé en argument. Dans cette suite, on remplace les multiple de 3 par «Fizz», les multiple de 5 par «Buzz» (et donc les multiples de 3 et 5 par «FizzBuzz»). On utilisera une boucle `for`.

```
$ ./fizzbuzz 16
1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz 16
```

Question 2.4 Faire un programme qui permet d'afficher un triangle d'une longueur qu'on précisera en argument. On utilisera une double boucle `for`. Par exemple, pour un argument de 7, le programme affichera :

```
$ ./triangle 7
#
# #
# # #
# # # #
# # # # #
# # # # # #
# # # # # # #
```