COMSM1201 : Exercises in C

Neill Campbell

Department of Computer Science, University of Bristol

Formatted in LaTeX, based on the Legrand Orange Book from BOOK-WEBSITE.COM

Contents

1. Hello World (Week 0 & 1)

The exercises in this Chapter are largely taken from the book "C by Dissection".

1.1 Lecture Notes Chapter B

Exercise 1.1 Once you've studied Chapter *B* of the lecture notes (*Hello World*), compile and run the examples given in the handout. ■

1.2 Twice the Sum

Here is part of a program that begins by asking the user to input three integers:

```
#include <stdio.h>

int main(void)
{
    int a, b, c;

    printf("Input three integers: ");
```

...

Exercise 1.2 Complete the program so that when the user executes it and types in 2, 3, and 7, this is what appears on the screen:

```
Input three integers: 2 3 7
Twice the sum of integers plus 7 is 31 !
```

■

1.3 Letter C

Execute this program so you understand the output:

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

```
#define HEIGHT 17

int main(void)
{

    int i = 0;

    printf("\n\nIIIIIII\n");
    while(i < HEIGHT){
        printf(" III\n");
        i = i + 1;
    }
    printf("IIIIIII\n\n\n");
    return 0;

}
```

Exercise 1.3 Write a similar program that prints a large letter C on the screen (it doesn't need to be curved!). ■

1.4 Lecture Notes Chapter C

Exercise 1.4 Once you've studied Chapter *C* of the lecture notes (*Grammar*), compile and run the examples given in the handout. ■

1.5 ++a++

Study the following code and write down what you think it prints.

```
int a, b = 0, c = 0;
a = ++b + ++c;
printf("%d %d %d\n", a, b, c);
a = b++ + c++;
printf("%d %d %d\n", a, b, c);
a = ++b + c++;
printf("%d %d %d\n", a, b, c);
a = b−− + −−c;
printf("%d %d %d\n", a, b, c);
```

Exercise 1.5 Then write a test program to check your answers. ■

1.6 Randomness

The function `rand()` returns values in the interval [0, RAND_MAX]. If we declare the variable `median` and initialise it to have the value `RAND_MAX/2`, then `rand()` will return a value that is sometimes larger than `median` and sometimes smaller.

Exercise 1.6 Write a program that calls `rand()`, say 500 times, inside a `for` loop, increments the variable `minus_cnt` every time `rand()` returns a value less than `median`. Each time through the `for` loop, print out the value of the difference of `plus_cnt` and `minus_cnt`. You might think that this difference should oscillate near zero. Does it ? ■

1.7 Lecture Notes Chapter D

Exercise 1.7 Once you've studied Chapter *D* of the lecture notes (*Flow Control*), compile and run the examples given in the handout. ■

1.8 find_max

Exercise 1.8 Write a program that finds the largest number entered by the user. Executing the program will produce something like:

```
How many numbers do you wish to enter ? 5
Enter 5 real numbers: 1.01 −3 2.2 7.0700 5
Maximum value: 7.07
```

■

1.9 Loving Oddness

Suppose that you detest even integers but love odd ones.

Exercise 1.9 Modify the `find_max` program so that all variables are of type `int` and that only odd integers are processed. Explain all this to the user via appropriate `printf()` statements. ■

1.10 Lecture Notes Chapter E

Exercise 1.10 Once you've studied Chapter *E* of the lecture notes (*Functions*), compile and run the examples given in the handout. ■

1.11 Hailstone

The next number in a hailstone sequence is $n/2$ if the current number n is even, or $3n+1$ if the current number is odd. If the initial number is 77, then the following sequence is produced:

```
77
232
116
58
29
88
44
22
11
34
```

Exercise 1.11 Write a program that, given a number typed by the user, prints out the sequence of *hailstone* numbers. The sequence terminates when it gets to 1. ■

1.12 Primes

A prime number can only be exactly divided by itself or 1. The number 17 is prime, but 16 is not because the numbers 2, 4 and 8 can divide it exactly. (Hint $16\%4 == 0$).

Exercise 1.12 Write a program that prints out the first n primes, where n is input by the user. The first 8 primes are:

```
2
3
5
7
11
13
7
19
```

What is the 3000^{th} prime ? ■

1.13 Triangles

A triangle can be equilateral (all three sides have the same length), isosceles (has two equal length sides), scalene (all the sides have a different length), or right angled where if the three sides are a, b and c, and c is the longest, then : $c = \sqrt{a^2 + b^2}$

Exercise 1.13 Write a program so that you can process a number of triples of side lengths in a single run of your program using a suitable unlikely input value for the first integer in order to terminate the program. e.g. -999.

Think hard about the test data for your program to ensure that all possible cases are covered and all invalid data results in a sensible error message. Such cases can include sides of negative length, and impossible triangles (e.g. one side is longer than the sum of the other two). ■

1.14 Lecture Notes Chapter F

Exercise 1.14 Once you've studied Chapter F of the lecture notes (*Data Storage*), compile and run the examples given in the handout. ■

1.15 Unit Circle

In mathematics, for all real x, it is true that:

$$sin^2(x) + cos^2(x) = 1$$

i.e. $sin(x) * sin(x) + cos(x) * cos(x) = 1$.

Exercise 1.15 Write a program to demonstrate this for values of x input by the user. ■

1.16 Time Flies

Exercise 1.16 Write a program which allows the user to enter two times in 24-hour clock format, and computes the length of time between the two, e.g.:

```
Enter two times : 23:00 04:15
Difference is : 5:15
```

or,

```
Enter two times : 23:40 22:50
Difference is : 23:10
```

■