Errata for

Programming, Problem Solving, and Abstraction with C

by Alistair Moffat

as at **August 24, 2014**

Chapter 1 Computers and Programs

page 3

In the line "by a factor of" there is a missing space between the word "million" and the hyphen.

page 11

In the first line, "more valuable that the hardware" should be "more valuable than the hardware".

Chapter 2 Numbers In, Numbers Out

page 23

In line two, from "As a slightly confusing" through to the end of that paragraph is incorrect. In fact, splitting a format control string over two or more `scanf` statements will yield identical behavior, *provided* that every component of the format control string still appears in the same order as when it was in a single `scanf`. So the example given is incorrect, and both options will give the same outcomes.

On the other hand, if the first option had been

```
scanf("%f %c",&x,&c)
```

with a blank between the two format descriptors, then it *would* behave differently to the subsequent two that are shown, because the blank "matches" any string of whitespace characters that might appear between the number and the character, and is not contained in either of the two `scanf`s that make up the second option.

Bottom line: whitespace skipping is not undertaken prior to a `%c` descriptor, regardless of the context. If whitespace skipping is required in order to read a non-whitespace character with a `%c` descriptor, a blank should be added to the format control string. Skipping of leading whitespace is always undertaken with numeric format descriptors.

page 26

In line four the code for "`d`" is 100, not 101.

page 28

The `printf` statement in the box in Exercise 2.4 refers to variables `n`, `x`, and `m` in the format control string, but actually prints variables `n`, `z`, and `m`. It is not "wrong" as it stands, but would certainly be less confusing if variable `x` was the second value printed, rather than `z`.

Chapter 4 Loops

page 52
Line 7, “more important that the particular style”, should say “more important than the particular style”.

page 57
The program in Figure 4.12 indicates that the user should use “control-D” to indicate the end of the input. With most Unix shells, input from the keyboard is indicated by typing a control-D character. This might differ when other operating systems are being used, or with other shells. In particular, in the Windows environment, you might find that you need to use control-Z rather than control-D. On most Unix shells, control-Z suspends the executing program, and resumes the shell.

Chapter 5 Functions

page 73
In the first line of text, “There is final point” should say “There is one final point”.

Chapter 6 Functions and Pointers

page 83
In the last line, “`EXIT_FAILURE` and `EXIT_SUCCESS` are defined to be zero and one respectively” should say “`EXIT_FAILURE` and `EXIT_SUCCESS` are defined to be one and zero respectively”.

page 90
In the second line of Section 6.5, “other functions in unnecessary” should say “other functions is unnecessary”.

Chapter 7 Arrays

page 111
In the second line of text, “But any type can used” should say “But any type can be used”.

page 111
In the last paragraph, in the second line the calculation “`&A([i])+4*(j-1)`” should be “`&A([i])+4*j`”; and in the third-to-last line the calculation “`A + 4*COLS*(i-1)`” should be “`A + 4*COLS*i`”.

Chapter 8 Structures

page 141
The paragraph that starts with “The third function” claims that all of the parentheses are required in the expression `&((*planet).mass)`, which accesses the address of the `mass` field of the object pointed at by `planet`. In fact, the “`.`” operator has higher precedence than does “`&`”, so it is sufficient to write `&(*planet).mass`. The same precedence helps with

the program in Figure 8.5 on page 140, where function `read_planet` can use `&planet->distance` rather than `&(planet->distance)`, and so on.

Chapter 9 Problem Solving

page 159
The asymmetry in the way the `if` guard is presented means that the `bisection` function does not converge if the situation is reached in which `f(mid)==0`; that is, if `mid` becomes the exact root by luck while the search is proceeding. To fix the problem, the code needs to deal with three distinct cases rather than two:

```
if (fx1*fmid < 0) {
    /* root is to left of middle */
    x2 = mid;
    fx2 = fmid;
} else if (fx1*fmid > 0) {
    /* root is to right of middle */
    x1 = mid;
    fx1 = fmid;
} else {
    x1 = x2 = mid;
}
```

page 161
Exercise 9.3, “Write a program that deals four random five-card poker hand” should be “Write a program that deals four random five-card poker hands”

Chapter 10 Dynamic Structures

page 165
In both parts of Figure 10.1 there is a right parenthesis missing after the word `char` in the string `sizeof(char`.

page 166
In Figure 10.2 it would probably have been polite to show the `#include <stdlib.h>` that has been trimmed off the top of the figure. Without the prototype for `malloc` that appears in the header file, you may get compiler warnings about type mismatches.

page 179
In line 2, the calculation $1.4 \log_2 n$ yields 28 when $n = 1{,}000{,}000$, and not 29 as is written.

page 180
Figure 10.10 contains an error in function `double_ascending`. The two `if` statements should read:

```
if (*d1<*d2) return -1;
if (*d1>*d2) return +1;
```

I made exactly the mistake that I warned you about on page 116. Oops.

page 184
: Figure 10.13 appears to be the only place where I have used argument variable names within the argument list of a function prototype. This is not incorrect, but does differ from the style used in the rest of the examples in the book.

page 188
: A `#include <string.h>` is required in the program in Figure 10.17.

Chapter 11 Files

page 194
: In Table 11.1, the entries for `fread` and `fwrite` should say "[reading/writing] objects of the size given by the second argument. The third argument indicates the number of objects". The argument order that I have indicated in the table is the `qsort` order, but `fread` and `fwrite` swap the middle two arguments. On page 197 I have warned against exactly this error, but have gone and made it anyway. Doh!

page 196
: In Figure 11.1, the comment prior to function `first_lines` has two `*/` closing combinations. One of them should be deleted.

Chapter 13 Everything Else

page 235
: The decimal equivalent of the last example shown in Table 13.4 should be negative, that is, $-.0999755859375$. and not .0999755859375 as claimed. The rounding error is not that big!