# *The C Programming Language: Chapter 1*

*Eric Bailey*

*March 4, 2018* [1]

[1] Last updated October 16, 2018

Write an abstract

*Contents*

## *Hello, world!*

Covers Exercises 1-1 and 1-2.

2a ⟨*hello.c* 2a⟩≡
⟨*Include the standard I/O functions.* 12b⟩

```
int main()
{
    printf("Hello, world!\n");
}
```

Uses printf 12b.
Root chunk (not used in this document).

## *Fahrenheit-Celsius table*

Covers Exercises 1-3, 1-4, and 1-5.

2b ⟨*fahrcels.c* 2b⟩≡
⟨*Include the standard I/O functions.* 12b⟩
⟨*Include the standard string functions.* 12c⟩

This definition is continued in chunks 2 and 3.
Root chunk (not used in this document).

Declare some useful constants.

2c ⟨*fahrcels.c* 2b⟩+≡

```
#define LOWER 0
#define UPPER 300
#define STEP  20
```

Defines:
LOWER, used in chunk 3b.
STEP, used in chunk 3b.
UPPER, used in chunk 3b.

## *Exercise 1-3*

2d ⟨*fahrcels.c* 2b⟩+≡

```
void print_header(char lhs[], char rhs[])
{
    printf("| %s | %s |\n", lhs, rhs);
    putchar('|');
    for (int i = -2; i < (int)strlen(lhs); ++i)
        putchar('-');
    putchar('+');
    for (int i = -2; i < (int)strlen(rhs); ++i)
        putchar('-');
    puts("|");
}
```

Defines:
print_header, used in chunk 3.
Uses printf 12b, putchar 12b, puts 12b, and strlen 12c.

### *Exercise 1-4*

3a ⟨*fahrcels.c* 2b⟩+≡

```
void celsfahr()
{
    print_header("Celsius", "Fahrenheit");
    for (int celsius = 0; celsius ≤ 300; celsius += 20)
        printf("| %7d | %10.0f |\n", celsius, 32.0 + (9.0/5.0) * celsius);
}
```

Defines:
`celsfahr`, used in chunk 3c.
Uses `printf` 12b and `print_header` 2d.

### *Exercise 1-5*

3b ⟨*fahrcels.c* 2b⟩+≡

```
void fahrcels()
{
    print_header("Fahrenheit", "Celsius");
    for (int fahr = UPPER; fahr ≥ LOWER; fahr -= STEP)
        printf("| %10d | %7.1f |\n", fahr, (5.0/9.0) * (fahr-32.0));
}
```

Defines:
`fahrcels`, used in chunk 3c.
Uses `LOWER` 2c, `STEP` 2c, `UPPER` 2c, `printf` 12b, and `print_header` 2d.

### *The* `main` *function*

3c ⟨*fahrcels.c* 2b⟩+≡

```
int main()
{
    fahrcels();
    puts("\n");
    celsfahr();

    return 0;
}
```

Uses `celsfahr` 3a, `fahrcels` 3b, and `puts` 12b.

## *Copy*

Covers Exercises 1-6 and 1-7.

4c ⟨*copy.c* 4c⟩≡
⟨*Include the standard I/O functions.* 12b⟩

```
int main()
{
    int c;
    ⟨For each character c until EOF 4a⟩
        ⟨Print the character. 4b⟩

    return 0;
}
```

Root chunk (not used in this document).

## *Character Counting*

4d ⟨*wc.c* 4d⟩≡
⟨*Include the standard I/O functions.* 12b⟩
⟨*Include the boolean type and values.* 12a⟩

This definition is continued in chunks 4, 5, and 7.
Root chunk (not used in this document).

4e ⟨*wc.c* 4d⟩+≡

```
double char_count()
{
    double nc;

    for (nc = 0; getchar() ≠ EOF; ++nc)
        ;

    return nc;
}
```

Defines:
char_count, never used.

4a ⟨*For each character* c *until* EOF 4a⟩≡

```
while ((c = getchar()) ≠ EOF)
```

This code is used in chunks 4–9 and 11.

4b ⟨*Print the character.* 4b⟩≡

```
putchar(c);
```

Uses putchar 12b.
This code is used in chunks 4c, 6a, and 7a.

### *Line Counting*

5b ⟨*wc.c* 4d⟩+≡
```
int line_count()
{
    int c, nl;

    nl = 0;
    ⟨For each character c until EOF 4a⟩
        if (⟨the character is a newline 5a⟩)
            ++nl;

    return nl;
}
```
Defines:
line_count, never used.

5a ⟨*the character is a newline* 5a⟩≡
```
c == '\n'
```
This code is used in chunks 5 and 7d.

### *Exercise 1-8*

For our purposes, whitespace is a space, tab, or newline.

5d ⟨*the character is whitespace* 5d⟩≡
```
c == ' ' || ⟨the character is a newline 5a⟩ || ⟨the character is a tab 5c⟩
```
This code is used in chunks 5e and 7–9.

5c ⟨*the character is a tab* 5c⟩≡
```
c == '\t'
```
This code is used in chunks 5d and 6d.

5e ⟨*wc.c* 4d⟩+≡
```
bool is_whitespace(int c)
{
    return (⟨the character is whitespace 5d⟩);
}
```
Defines:
is_whitespace, used in chunk 5f.
Uses bool 12a.

5f ⟨*wc.c* 4d⟩+≡
```
double ws_count()
{
    double ns = 0;
    int c = 0;

    ⟨For each character c until EOF 4a⟩
        if (is_whitespace(c))
            ++ns;

    return ns;
}
```
Defines:
ws_count, never used.
Uses is_whitespace 5e.

### *Exercise 1-9*

6a ⟨*catblanks.c* 6a⟩≡
⟨*Include the standard I/O functions.* 12b⟩
⟨*Include the boolean type and values.* 12a⟩

```
int main()
{
    int c;
    bool prev_blank = false;

    ⟨For each character c until EOF 4a⟩ {
        if (!(prev_blank && c == ' '))
            ⟨Print the character. 4b⟩
        prev_blank = (c == ' ');
    }

    return 0;
}
```

Uses bool 12a.
Root chunk (not used in this document).

### *Exercise 1-10*

Process each character c.

6c ⟨*unambiguous.c* 6b⟩+≡
```
int c;

⟨For each character c until EOF 4a⟩ {
```

Replace each tab by \t.

6d ⟨*unambiguous.c* 6b⟩+≡
```
if (⟨the character is a tab 5c⟩)
    fputs("\\t", stdout);
```

Uses fputs 12b and stdout 12b.

Replace each backspace by \b.

6f ⟨*unambiguous.c* 6b⟩+≡
```
else if (⟨the character is a backspace 6e⟩)
    fputs("\\b", stdout);
```

Uses fputs 12b and stdout 12b.

Replace each backslash by \\.

6h ⟨*unambiguous.c* 6b⟩+≡
```
else if (⟨the character is a backslash 6g⟩)
    fputs("\\\\", stdout);
```

Uses fputs 12b and stdout 12b.

6b ⟨*unambiguous.c* 6b⟩≡
⟨*Include the standard I/O functions.* 12b⟩

```
int main()
{
```

This definition is continued in chunks 6 and 7.
Root chunk (not used in this document).

6e ⟨*the character is a backspace* 6e⟩≡
```
c == '\b'
```

This code is used in chunk 6f.

6g ⟨*the character is a backslash* 6g⟩≡
```
c == '\\'
```

This code is used in chunk 6h.

Otherwise print the character unchanged.

7a ⟨*unambiguous.c* 6b⟩+≡
```
        else
            ⟨Print the character. 4b⟩
```

## *Word Counting*

7c ⟨*wc.c* 4d⟩+≡
```
#define IN  1
#define OUT 0
```
Defines:
IN, used in chunks 7–9.
OUT, used in chunks 7–9.

7d ⟨*wc.c* 4d⟩+≡
```
int main()
{
    int c, nl, nw, nc, state;

    state = OUT;
    nl = nw = nc = 0;
    ⟨For each character c until EOF 4a⟩ {
        ++nc;
        if (⟨the character is a newline 5a⟩)
            ++nl;
        if (⟨the character is whitespace 5d⟩)
            state = OUT;
        else if (state == OUT) {
          state = IN;
          ++nw;
        }
    }

    printf("%7d%8d%8d\n", nl, nw, nc);

    return 0;
}
```
Uses IN 7c, OUT 7c, and printf 12b.

Finally, close the **while** loop and exit.

7b ⟨*unambiguous.c* 6b⟩+≡
```
    }

    return 0;
}
```

*Exercise 1-12*

8 ⟨*words.c* 8⟩≡
⟨*Include the standard I/O functions.* 12b⟩

```
#define IN    1
#define OUT   0


int main()
{
    int c, state;

    state = OUT;
    ⟨For each character c until EOF 4a⟩ {
        if (⟨the character is whitespace 5d⟩) {
            if (state == IN)
                putchar('\n');
            state = OUT;
        } else {
            state = IN;
        }

        if (state == IN)
            putchar(c);
    }


    return 0;
}
```

Uses IN 7c, OUT 7c, and putchar 12b.
Root chunk (not used in this document).

*Arrays*

*Exercise 1-13*

Vertical histogram

9 ⟨*wordlength.c* 9⟩≡
⟨*Include the standard I/O functions.* 12b⟩

```
#define IN    1
#define OUT   0

#define MAX_WORD_LENGTH 10
#define TERM_WIDTH 80


int main()
{
    int c, state, wl;
    int length[MAX_WORD_LENGTH+1];

    for (int i = 0; i ≤ MAX_WORD_LENGTH; ++i)
        length[i] = 0;

    state = OUT;
    wl = 0;
    ⟨For each character c until EOF 4a⟩ {
        if (⟨the character is whitespace 5d⟩) {
            if (state == IN) {
                state = OUT;
                ++length[wl ≤ MAX_WORD_LENGTH ? wl-1 : MAX_WORD_LENGTH];
            }
        } else {
            if (state == OUT) {
                state = IN;
                wl = 0;
            }
            ++wl;
        }
    }

    for (int j = 0; j ≤ MAX_WORD_LENGTH; ++j) {
        if (j == MAX_WORD_LENGTH)
            printf(">%d: ", MAX_WORD_LENGTH);
        else
            printf(" %2d: ", j+1);

        for (int k = 0; k < length[j]; ++k)
            putchar('#');
        putchar('\n');
    }
```

```
    return 0;
}
```

Uses IN 7c, OUT 7c, printf 12b, and putchar 12b.
Root chunk (not used in this document).

## *Exercise 1-14*

10a ⟨*charfreq.c* 10a⟩≡

```
⟨Include the standard I/O functions. 12b⟩


#define MIN_ASCII 0
#define MAX_ASCII 0177
```

This definition is continued in chunks 10b and 11.
Root chunk (not used in this document).

10b ⟨*charfreq.c* 10a⟩+≡

```
void prchar(int c)
{
    switch (c) {
        case ' ':
            printf("%11s", "<space>");
            break;
        case '\b':
            printf("%11s", "<backspace>");
            break;
        case '\n':
            printf("%11s", "<newline>");
            break;
        case '\t':
            printf("%11s", "<tab>");
            break;
        default:
            /* FIXME: why can't I return this? */
            /* return ((char[2]) { (char) c, '\0' }); */
            printf("%11c", c);
            break;
    }
}
```

Defines:
  prchar, used in chunk 11.
Uses printf 12b.

11 ⟨*charfreq.c* 10a⟩+≡

```
int main()
{
    int c;
    int freq[MAX_ASCII+1] = {0};

    ⟨For each character c until EOF 4a⟩
        ++freq[c];

    for (int i = 0; i ≤ MAX_ASCII; ++i) {
        if (!freq[i]) continue;

        prchar(i);
        fputs(": ", stdout);
        for (int j = 0; j < freq[i]; ++j)
            putchar('#');
        putchar('\n');
    }


    return 0;
}
```

Uses fputs 12b, prchar 10b, putchar 12b, and stdout 12b.

*Functions*

Exercise 1-15

Exercise 1-16

*Character Arrays*

Exercise 1-17

Exercise 1-18

Exercise 1-19

*External Variables and Scope*

Exercise 1-20

Exercise 1-21

Exercise 1-22

Exercise 1-23

Exercise 1-24

*Common Headers*

12a ⟨*Include the boolean type and values.* 12a⟩≡

```
#include <stdbool.h>
```

Defines:
`bool`, used in chunks 5e and 6a.
This code is used in chunks 4d and 6a.

12b ⟨*Include the standard I/O functions.* 12b⟩≡

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

Defines:
`fputs`, used in chunks 6 and 11.
`printf`, used in chunks 2, 3, 7d, 9, and 10b.
`putchar`, used in chunks 2d, 4b, 8, 9, and 11.
`puts`, used in chunks 2d and 3c.
`stdout`, used in chunks 6 and 11.
This code is used in chunks 2, 4, 6, and 8–10.

12c ⟨*Include the standard string functions.* 12c⟩≡

```
#include <string.h>
```

Defines:
`strlen`, used in chunk 2d.
This code is used in chunk 2b.

## *Chunks*



## *Index*

ws_count: 5f