

Princeton University

COS 217: Introduction to Programming Systems

Writing Binary Data

Example 1

We wish to write the long 0 to a file named "data" exactly as it would appear in memory as an eight-byte entity. That is, we wish to write these eight bytes to the file:

```
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```

Open the File

```
FILE *psFile;  
psFile = fopen("data", "w");
```

Attempt 1 (Incorrect)

```
fprintf(psFile, "00000000"); /* Writes 00110000 00110000 00110000 00110000  
                                00110000 00110000 00110000 00110000 */
```

Attempt 2 (Incorrect)

```
fprintf(psFile, "%ld", 0); /* Writes 00110000 */
```

Attempt 3 (Incorrect)

```
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */  
fprintf(psFile, "%c", '0'); /* Writes 00110000 */
```

Attempt 4 (Incorrect)

```
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */  
putc('0', psFile); /* Writes 00110000 */
```

Attempt 5 (Correct)

```
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */  
fprintf(psFile, "%c", 0); /* Writes 00000000 */
```

Attempt 6 (Correct)

```
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */  
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
```

Attempt 7 (Correct)

```

putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */

```

Attempt 8 (Correct)

```

putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */

```

Attempt 9 (Correct)

<--- the preferred approach

```

long lData;
...
lData = 0;
fwrite(&lData, sizeof(long), 1, psFile); /* Writes 00000000 00000000 00000000 00000000
                                         00000000 00000000 00000000 00000000 */

```

Close the File

```
fclose(psFile);
```

Example 2

We wish to write the long 5678 to a file named "data" exactly as it would appear in memory as an eight-byte entity. As humans, we would express the integer 5678 in binary like this:

```

00000000 00000000 00000000 00000000 00000000 00000000 00101110 00101110
most sig                                     least sig
byte                                          byte

```

But remember that Intel is a little-endian computer. In the memory of a little-endian computer, the least significant byte of an integer is in the lowest memory location. So the integer 5678 appears in memory like this:

```

00101110 00101110 00000000 00000000 00000000 00000000 00000000 00000000
least sig                                     most sig
byte                                          byte

```

Or, more precisely, like this:

```

pretend
address
1000    00101110  least sig byte
1001    00101110
1002    00000000
1003    00000000
1004    00000000
1005    00000000
1006    00000000
1007    00000000  most sig byte

```

Open the File

```
FILE *psFile;
psFile = fopen("data", "w");
```

Attempt 1 (Incorrect)

```
fprintf(psFile, "5678"); /* Writes 00110101 00110110 00110111 00111000 */
```

Attempt 2 (Incorrect)

```
fprintf(psFile, "%d", 5678); /* Writes 00110101 00110110 00110111 00111000 */
```

Attempt 3 (Incorrect)

```
fprintf(psFile, "%c", '5'); /* Writes 00110101 */
fprintf(psFile, "%c", '6'); /* Writes 00110110 */
fprintf(psFile, "%c", '7'); /* Writes 00110111 */
fprintf(psFile, "%c", '8'); /* Writes 00111000 */
```

Attempt 4 (Incorrect)

```
putc('5', psFile); /* Writes 00110101 */
putc('6', psFile); /* Writes 00110110 */
putc('7', psFile); /* Writes 00110111 */
putc('8', psFile); /* Writes 00111000 */
```

Attempt 5 (Correct)

```
fprintf(psFile, "%c", 46); /* Writes 00101110 */
fprintf(psFile, "%c", 22); /* Writes 00010110 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
fprintf(psFile, "%c", 0); /* Writes 00000000 */
```

Attempt 6 (Correct)

```
fprintf(psFile, "%c", 0x2e); /* Writes 00101110 */
fprintf(psFile, "%c", 0x16); /* Writes 00010110 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
```

Attempt 7 (Correct)

```
putc(46, psFile); /* Writes 00101110 */
putc(22, psFile); /* Writes 00010110 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
putc(0, psFile); /* Writes 00000000 */
```

Attempt 8 (Correct)

```
putc(0x2e, psFile); /* Writes 00101110 */
putc(0x16, psFile); /* Writes 00010110 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
```

Attempt 9 (Correct)

```
long lData;
...
lData = 5678;
fwrite(&lData, sizeof(long), 1, psFile); /* Writes 00101110 00010110 00000000 00000000
00000000 00000000 00000000 00000000 */
```

Close the File

```
fclose(psFile);
```

Copyright © 2015 by Robert M. Dondero, Jr.