Princeton University
COS 217: Introduction to Programming Systems
Writing Binary Data

Example 1

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

00000000 00000000 00000000 00000000

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

Attempt 1 (Incorrect)
```
fprintf(psFile, "0000"); /* Writes 00110000 00110000 00110000 00110000 */
```

Attempt 2 (Incorrect)
```
fprintf(psFile, "%d", 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 */
```

Attempt 4 (Incorrect)
```
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 */
```

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 */
```

Attempt 7 (Correct)
```
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 */
```

Attempt 9 (Correct) <--- the preferred approach
```
unsigned int uiData;
...
uiData = 0;
fwrite(&uiData, 4, 1, psFile); /* Writes 00000000 00000000 00000000 00000000 */

/* Uses the standard C fwrite() function:
   size_t fwrite(const void *p, size_t s, size_t n, FILE *stream);
   Writes p to stream.  p is an array of n elements, each of size s. */
```

Attempt 10 (Correct, but Non-Standard)
```
putw(0, psFile);  /* Writes 00000000 00000000 00000000 00000000 */

/* Uses the non-standard putw() function:
   int putw(int w, FILE *stream); */
```

Close the File
```
fclose(psFile);
```

Example 2

We wish to write the integer 1234 to a file named "data" exactly as it would appear in memory as a four-byte entity. Note 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 we wish to write these four bytes to the file:

11010010 00000100 00000000 00000000

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

Attempt 1 (Incorrect)
```
fprintf(psFile, "1234"); /* Writes 00110001 00110010 00110011 00110100 */
```

Attempt 2 (Incorrect)
```
fprintf(psFile, "%d", 1234); /* Writes 00110001 00110010 00110011 00110100 */
```

Attempt 3 (Incorrect)
```
fprintf(psFile, "%c", '1');  /* Writes 00110001 */
fprintf(psFile, "%c", '2');  /* Writes 00110010 */
fprintf(psFile, "%c", '3');  /* Writes 00110011 */
fprintf(psFile, "%c", '4');  /* Writes 00110100 */
```

Attempt 4 (Incorrect)
```
putc('1', psFile);  /* Writes 00110001 */
putc('2', psFile);  /* Writes 00110010 */
putc('3', psFile);  /* Writes 00110011 */
putc('4', psFile);  /* Writes 00110100 */
```

Attempt 5 (Correct)

```
fprintf(psFile, "%c", 210); /* Writes 11010010 */
fprintf(psFile, "%c", 4);   /* Writes 00000100 */
fprintf(psFile, "%c", 0);   /* Writes 00000000 */
fprintf(psFile, "%c", 0);   /* Writes 00000000 */
```

Attempt 6 (Correct)

```
fprintf(psFile, "%c", 0xd2); /* Writes 11010010 */
fprintf(psFile, "%c", 0x04); /* Writes 00000100 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
fprintf(psFile, "%c", 0x00); /* Writes 00000000 */
```

Attempt 7 (Correct)

```
putc(210, psFile); /* Writes 11010010 */
putc(4, psFile);   /* Writes 00000100 */
putc(0, psFile);   /* Writes 00000000 */
putc(0, psFile);   /* Writes 00000000 */
```

Attempt 8 (Correct)

```
putc(0xd2, psFile); /* Writes 11010010 */
putc(0x04, psFile); /* Writes 00000100 */
putc(0x00, psFile); /* Writes 00000000 */
putc(0x00, psFile); /* Writes 00000000 */
```

Attempt 9 (Correct) <--- the preferred approach

```
unsigned int uiData;
...
uiData = 1234;
fwrite(&uiData, 4, 1, psFile); /* Writes 11010010 00000100 00000000 00000000 */

/* Uses the standard C fwrite() function:
   size_t fwrite(const void *p, size_t s, size_t n, FILE *stream);
   Writes p to stream.  p is an array of n elements, each of size s. */
```

Attempt 10 (Correct, but Non-Standard)

```
putw(1234, psFile);  /* Writes 11010010 00000100 00000000 00000000 */

/* Uses the non-standard putw() function:
   int putw(int w, FILE *stream); */
```

Close the File

```
fclose(psFile);
```