最終更新日: 2019 年 8 月 5 日

「やさしく学べる C 言語入門 [第 2 版]]」(第 3,4 刷) 正誤表

	誤	正
p.2	**バイト**　8 ビットの集まり．単位を byte や B で表す.	**バイト**　複数ビットのこと．一般には 1 バイトは 8 ビットで，単位を byte や B で表す.
p.19, 9 行目	なお，3 行目の `int main{` と 10 行目の `}` との間に記述されたプログラムを	なお，3 行目の `int main(void){` と 10 行目の `}` との間に記述されたプログラムを
p.47, 4 行目	通常は，１つの整数型の変数について場合分けを行います.	通常は，１つの整数型の定数について場合分けを行います.
p.50,「実行例」の後	なお，switch 文は整数型の変数について場合分け	なお，switch 文は整数型の定数について場合分け
p.78, 5.4 節	**5.4 多次元配列*** 3 つ以上の添字を持つ配列を**多次元配列**といいますが，この多次元配列も，例えば `double a[4][5][6]` のように宣言することができます．C 言語では，コンパイラが許す限り多次元配列を宣言することができるのですが，一般に多次元配列を利用するとプログラムの実行速度がかなり遅くなるのでほとんど使われません.	**5.4 多次元配列*** 3 つ~~以上~~の添字を持つ配列を **3 次元配列**といいますが，この 3 次元配列も，例えば `double a[4][5][6]` のように宣言することができます．なお，2 次元以上の配列を総称して**多次元配列**といいます．C 言語では，コンパイラが許す限り多次元配列を宣言~~することが~~できるのですが，一般に 3 次元以上の多次元配列を利用するとプログラムの実行速度がかなり遅くなるので，2 次元配列までがよく使われます.

<table>
<tr><th></th><th>誤</th><th>正</th></tr>
<tr><td>p.91, 下から 5 行目</td><td><code>HelloWorld(void);</code></td><td><code>HelloWorld();</code></td></tr>
<tr><td>p.105, 中程</td><td>通常，ポインタ変数は NULL で初期化する習慣になっており，例えば，</td><td>通常，ポインタ変数は NULL で初期化する習慣になっており，それを明示したいときは, 例えば，</td></tr>
<tr><td>p.105，プログラム 7.3 の前</td><td>ただし，ポインタ変数を NULL で初期化したままで，何らかの変数のアドレスを指定しなかった場合，</td><td>ただし，ポインタ変数を NULL で初期化したままで，ポインタが指す変数の領域を確保しなかった場合，</td></tr>
<tr><td>p.105，プログラム 7.3</td><td><code>int *p; /* ポインタ変数 p を NULL で初期化 */</code>
<code>*p = 10; /* アドレスを指定しないまま値を代入 */</code></td><td><code>int *p = NULL ; /* ポインタ変数 p を NULL で初期化 */</code>
<code>*p = 10; /* 領域を確保しないまま値を代入 */</code></td></tr>
<tr><td>p.106, 表 7.1 の先頭行</td><td><table><tr><th></th><th>通常の変数</th><th>ポインタ変数</th><th>例</th></tr></table></td><td><table><tr><th></th><th>通常の変数</th><th>ポインタ変数</th><th>説明</th></tr></table></td></tr>
<tr><td>p.110, 3 行目</td><td>ここで登場する sizeof 関数 (sizeof はサイズオブと読む) は配列全体の大きさを求めるための関数です.</td><td>ここで登場する sizeof 演算子 (sizeof はサイズオブと読む) は配列全体の大きさを求めるための演算子です.</td></tr>
<tr><td>p. 256, さくいん</td><td>sizeof 関数 110</td><td>sizeof 演算子 110</td></tr>
<tr><td>p.117 の脚注</td><td>例えば，標準入力ストリーム (standard input stream) はキーボードからプログラムへ入力されるデータの流れ，標準出力ストリーム (standard output stream) はプログラムからディスプレイ画面に出力されるデータの流れを指します.</td><td>左記の記述をすべて削除.
【削除する理由】
読者がイメージしやすように，標準入力を「キーボード」，標準出力を「ディスプレイ画面」と表記した．しかし，実際には，これらはファイルなど別のものに指定できるため，必ずしも「キーボード」や「ディスプレイ画面」ではない．分かりやすやよりも，誤解を与える悪影響の方が大きいと判断した.</td></tr>
</table>

<table>
<tr><th></th><th>誤</th><th>正</th></tr>
<tr><td>p.227，表の「関数の内部」</td><td>
<table>
<tr><td>関数の内部</td><td>static</td><td>最初の関数呼び出し時に確保，その後も不変</td><td>宣言されている関数内</td></tr>
</table>
</td><td>
<table>
<tr><td>関数の内部</td><td>static</td><td>プログラム実行時に確保，その後も不変</td><td>宣言されている関数内</td></tr>
</table>
</td></tr>
</table>