C言語の基礎1

C言語

> https://ja.wikipedia.org/wiki/C%E8%A8%80%E8%AA%9E

歴史

誕生

C言語は、AT&Tベル研究所のケン・トンプソンが開発したB言語の改良として誕生した（#外部リンクの「The Development of the C Language」参照）。

1972年、トンプソンとUNIXの開発を行っていたデニス・リッチーはB言語を改良し、実行可能な機械語を直接生成するC言語のコンパイラを開発した[7]。後に、UNIXは大部分をC言語によって書き換えられ、C言語のコンパイラ自体も移植性の高い実装のPortable C Compilerに置き換わったこともあり、UNIX上のプログラムはその後にC言語を広く利用するようになった。

ちなみに、「UNIXを開発するためにC言語が作り出された」と言われることがあるが、「The Development of the C Language」によると、これは正しくなく、経緯は以下の通りである。C言語は、当初はあくまでもOS上で動くユーティリティを作成する目的で作り出されたものであり、OSのカーネルを記述するために使われるようになるのは後の展開である。

- UNIXの開発当初、Multicsプロジェクトが目指していた高級言語によるOSの開発という目標は見送られた。
- アセンブリ言語でUNIXが作成されると、OS上で動くユーティリティを作成するためのプログラミング言語が必要とされた。
- ケン・トンプソンは、当初Fortranコンパイラを作ろうとしたが、途中で放棄し、新しい言語であるB言語を作成した。
- B言語はインタプリタ言語であったため動作が遅く、B言語でユーティリティを作ることはあまりなかった。
- B言語の欠点を解消するため、1971年に改良作業を開始した。
- 1972年にC言語のコンパイラができあがり、UNIXバージョン2において、いくつかのユーティリティを作成するために使用された。

C言語リファレンス

> https://ja.cppreference.com/w/c/language

C のキーワード

これは C の予約されているキーワードの一覧です。 これらは言語によって使用されるため、これらのキーワードは再定義することはできません。

auto **break** **case** **char** **const** **continue** **default** **do** **double** **else** **enum** **extern**	**float** **for** **goto** **if** **inline (C99以上)** **int** **long** **register** **restrict (C99以上)** **return** **short**	**signed** **sizeof** **static** **struct** **switch** **typedef** **union** **unsigned** **void** **volatile** **while**	***Alignas (C11以上)*** **Alignof (C11以上)** ***Atomic (C11以上)*** **Bool (C99以上)** ***Complex (C99以上)*** **Generic (C11以上)** ***Imaginary (C99以上)*** **Noreturn (C11以上)** ***Static_assert (C11以上)*** **Thread_local (C11以上)**

C言語の構文の基礎

.gitignore

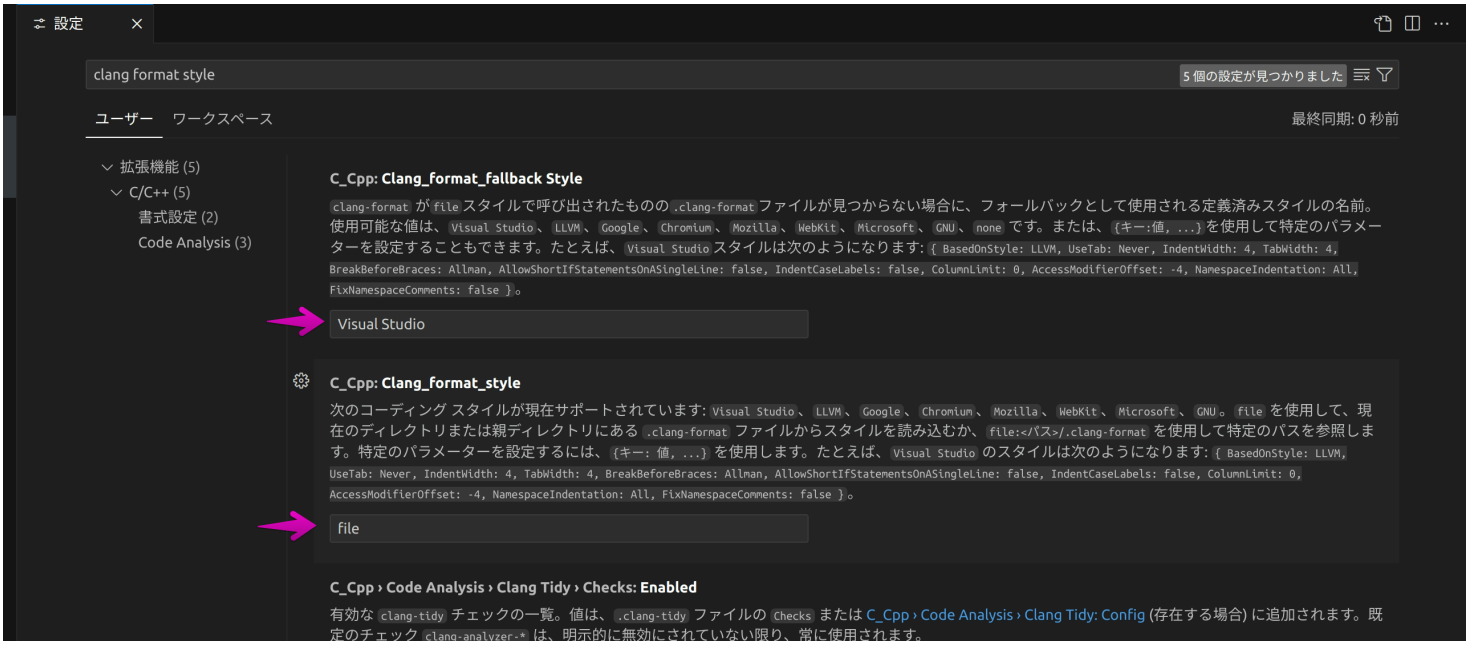
- C言語用の最低限の.gitignoreの雛形

```
# 拡張子なしのファイルを除外
*
!*/
!*.*

# 実行ファイル等を除外
*.out
*.o
```

formatterの設定

> https://www.kumikomist.com/tool/clangformat/
>
> https://qiita.com/grinpeaceman/items/04b6a973d11442a7ae48



【dot.clang_format】

- dotを削除し、`.clang_format` で保存する。
  - 保存場所：カレントディレクトリもしくは、その親ディレクトリ、ホームディレクトリ（例：/home/yoshimura/）

【最低限の設定】vscode

```
{
  BasedOnStyle: LLVM,
  IndentWidth: 4,
  TabWidth: 4,
  BreakBeforeBraces: Attach,
  AllowShortIfStatementsOnASingleLine: false,
  IndentCaseLabels: false,
  ColumnLimit: 0,
  AccessModifierOffset: -4,
  NamespaceIndentation: All,
  FixNamespaceComments: false}
```

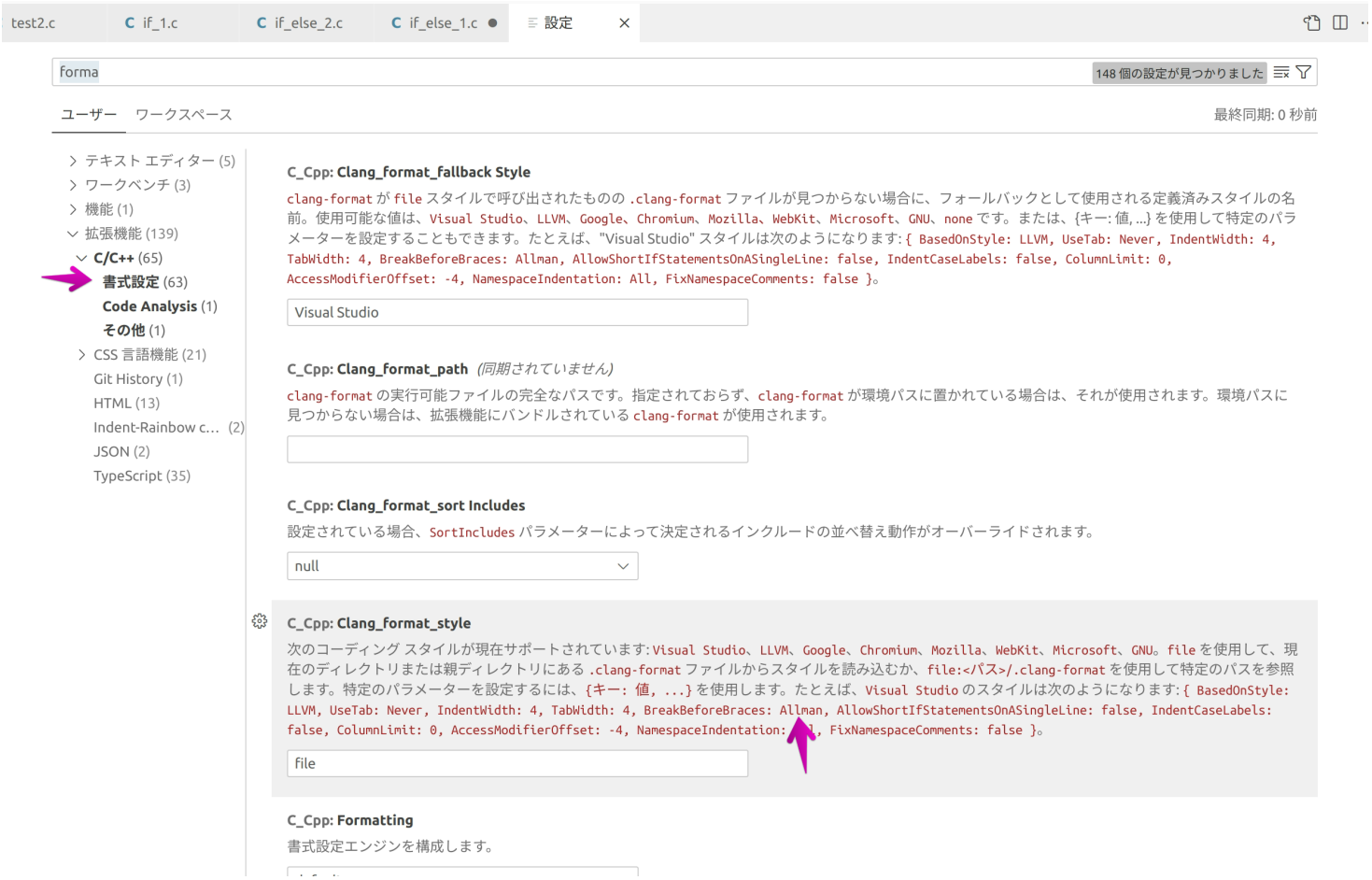
- Defaultと異なるのは以下の部分

```
BreakBeforeBraces: Attach,
```

- Defaultの設定

```
BreakBeforeBraces: Allman,
```

- vscodeの設定場所（ヒント）



一般的によく見る基本構文

【hello.c】

```
#include <stdio.h>

void main() {
    printf("Hello World.\n");
}
```

コンパイル＆リンクの仕方1

```
$ gcc hello.c
$ ls
a.out  hello.c
```

実行の仕方

```
$ ./a.out
Hello World.
```

現在の一般的な書き方

【hello_std.c】

```c
#include <stdio.h>

int main(int argc, char const *argv[]) {
    printf("Hello World.\n");
    return 0;
}
```

コンパイル＆リンクの仕方2

- 出力ファイル名を指定する

```
$ gcc hello_std.c -o hello2
$ ls
a.out  hello.c  hello2  hello_std.c
```

実行の仕方

```
$ ./hello2
```

変数とデータ型

変数の宣言と初期化

C言語で用いられるデータ型

- 以下の扱える範囲を調査してください。

データ型	説明	扱える範囲
char	１バイトの符号付整数。ASCIIコードといった文字コードに使用。	-128〜+127
unsigned char	１バイトの符号なし整数。	
short	2バイトの符号付整数。	
unsigned short	2バイトの符号なし整数。	
long	4バイトの符号付整数。	
unsigned long	4バイトの符号なし整数。	
int	2または4バイトの符号付整数。(コンパイラに依存)	
unsigned	2バイトまた4バイトの符号なし整数。(コンパイラに依存)	
float	4バイトの単精度浮動小数点実数。	
double	8バイトの倍精度浮動小数点実数。	

【var.c】

```c
#include <stdio.h>

int main(int argc, char const *argv[]) {
    int a;          // aを宣言
    a     = 1;      // aに1を代入
    int b = 1;      // bを宣言し、1で初期化
    int c, d = 1;   // cとdを宣言し、dのみ1で初期化

    return 0;
}
```

- 変数名にキーワード（予約語）は使用できない
- 使用できる文字
    - 半角の英文字（A〜Z，a〜z)、数字（0〜9)、アンダーバー（_）
- 変数名の最初の文字を数字にすることは出来ない。必ず英文字およびアンダーバーからはじめる。

- 英文字の大文字と小文字は別の文字として扱われる。

練習

- 上記データ型の整数に関して、扱える範囲を超えた場合にどうなるかを試すプログラムを作成してください。
    - 出力は、フォーマット書式指定子を調査の上、`printf()`関数を使用してください。
    - プログラム名は、over_maxmin.cとする

> https://www.k-cube.co.jp/wakaba/server/format.html

主な代入演算子

演算子	使用例	意味
+=	a+=1;	a=a+1;
-=	a-=1;	a=a-1;
*=	a*=2;	a=a*2;
/=	a/=2;	a=a/2;
%=	a%=2;	a=a%2;

データ型の変換

自動型変換

- 異なる型への代入は、以下のルールに従う
    - 精度の高い方（左の方）に変換される

```
double型 > float型 > long (int)型 > int型 > short (int)型 > char型
```

明示的型変換(キャスト)について

- 変数の前に `()` を付けて、変換したい型を記述する

【cast.c】

```c
#include <stdio.h>

int main(int argc, char const *argv[]) {
    int   num1 = 10;
    int   num2 = 3;
    float result1;
    int   result2;

    // int -> float にキャストして除算
    result1 = (float)num1 / num2;
    printf("result1: %f\n", result1);

    // float -> int にキャストして代入
    result2 = (int)result1;
    printf("result2: %d\n", result2);

    return 0;
}
```

- どのような結果が出力されるか、考えてください。

符号ありと符号なし

【cast2.c】

```c
#include <stdio.h>

int main(int argc, char const *argv[]) {
    char          chr = -100;
    unsigned char unchr;

    unchr = (unsigned char)chr;
    printf("unsigned char unchr : %u\n", unchr);

    return 0;
}
```

- どのような結果が出力されるか、考えてください。

解説（考え方）

step	手順	値・二進数
1	符号あり	-100
2	まず+100を二進数にする	
3	1の補数表現に変換する	
4	2の補数表現に変換する（+1する）	
5	そのまま十進数に変換する	