4 章_定石集

コードを用いた解説が多めなのでかなり端折ってます。すいません。

4-1 基本的な使い方

4-1-1 戻り値以外の方法で値を返してもらう

戻り値以外の方法で関数から型 T の値を返したいのなら、「T へのポインタ」を引数で返す。

4-1-2 配列を関数の引数として渡す

C では本当は「配列を引数として渡す」ことはできない。 しかし、配列の先頭要素へのポインタを渡すことで、あたかも配列を渡しているかのように扱うことができる。

型 T の配列を引数として渡したいなら「T へのポインタ」を渡せば良い。 ただし、配列の要素数は呼び出され側ではわからないので、必要なら別途渡すこと。

4-1-3 動的配列 - malloc()による可変長の配列

型 T の可変長の配列が欲しければ「T へのポインタ」を使い、領域を `malloc()` で動的に確保すれば良い。 ただし、配列の要素数は、別に管理する必要がある。

4-2 組み合わせて使う

4-2-1 動的配列の配列

コードを書いて使用例を解説している

4-2-2 動的配列の動的配列

コードを書いて使用例を解説している

4-2-3 コマンド行引数

コマンド行からプログラムを実行する時に、**コマンド行引数**として引数を与えることができる。

4-2-4 引数経由でポインタを返してもらう

コードを書いて使用例を解説している

> 例外処理では goto を使うとスッキリ書けることが多い

4-2-5 多次元配列を関数の引数として渡す

`4-1-2` にあったように、「T のポインタ」を渡せば良い

4-2-6 多次元配列を関数の引数として渡す(VLA 版)

`array[i][j]` の `j` の最大値を可変にできる。

4-2-7 縦横可変の 2 次元配列を malloc()で確保する(C99)

オセロの盤面のような 2 次元配列を、コードを書いて使用例を解説している

4-2-8 配列の動的配列

折れ線のつなぐ 2 次元座標の x,y 要素の配列からなる要素数が動的な配列についての宣言のはなし。 次の項につづく

4-2-9 変に凝る前に、構造体の使用を考えよう

```
typedef struct {
    double x;
    double y;
} Point;

typedef struct {
    int npoints;
    Point *point;
} Poliline;
```

とした方が、配列を使うよりスッキリする。

4-2-10 可変長構造体(ANSI C 版)

構造体の最後のメンバに限り、ポインタを介さずに、直接、可変長の配列を格納することができる。 このやりかたは ANSI C では反則技とされているが、大抵の環境で使える有効な手段。

4-2-11 フレキシブル配列メンバ(C99)

先程のテクニックは C99 で「フレキシブル配列メンバ」として正式に言語仕様で採用される。