変数とポインタ

プログラマのためのC言語 第12回

ポインタってなぁに？

point = ある空間の中の位置や場所

ポインタってなぁに？

point = ある空間の中の位置や場所

point + er = 場所や位置を表す者 = ポインタ

ポインタってなぁに？

point = ある空間の中の位置や場所

point + er = 場所や位置を表す者 = ポインタ

C言語のポインタ = メモリの位置や場所を表す

ポインタってなぁに？

point = ある空間の中の位置や場所

point + er = 場所や位置を表す者 = ポインタ

C言語のポインタ = メモリの位置や場所を表す

ポインタ = メモリのアドレスを入れた変数

ポインタってなぁに？

point = ある空間の中の位置や場所

point + er = 場所や位置を表す者 = ポインタ

C言語のポインタ = メモリの位置や場所を表す

ポインタ = メモリのアドレスを入れた変数

値	0	0	0	0	0	0	0	0	0
番地	10	11	12	13	14	15	16	17	18

p

※この図は簡易的なイメージです

ポインタってなぁに？

point = ある空間の中の位置や場所

point + er = 場所や位置を表す者 = ポインタ

C言語のポインタ = メモリの位置や場所を表す

ポインタ = メモリのアドレスを入れた変数

※この図は簡易的なイメージです

ポインタの基本的な事

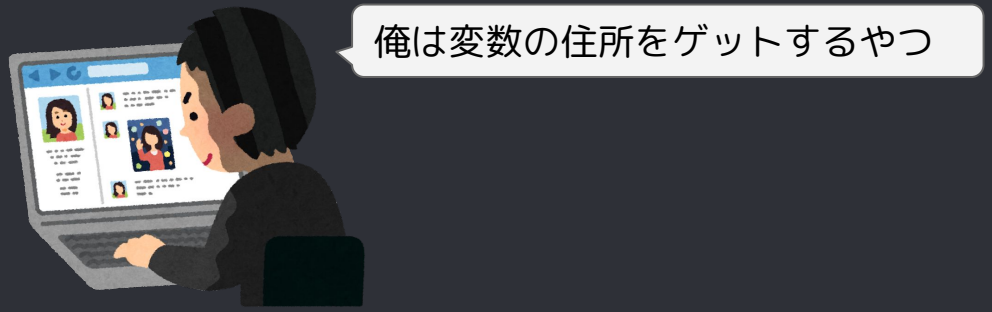
- ✔ **ポインタを定義する**
- ✔ ポインタに変数のアドレスを入れる
- ✔ ポインタが指すメモリの値を書き換える
- ✔ ぬるぽ
- ✔ 適当なアドレスが入ったポインタ

```
int*   i;
char*  c;
float* f;
```

変数を定義するとき
型の後ろに*を付けるとポインタって意味になる

ポインタの基本的な事

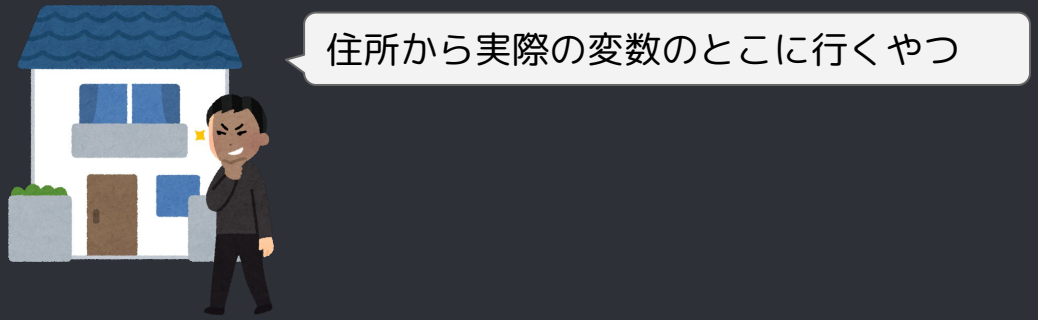
- ✔ ポインタを定義する
- ✔ **ポインタに変数のアドレスを入れる**
- ✔ ポインタが指すメモリの値を書き換える
- ✔ ぬるぽ
- ✔ 適当なアドレスが入ったポインタ



& アドレス演算子

ポインタの基本的な事

- ✔ ポインタを定義する
- ✔ ポインタに変数のアドレスを入れる
- ✔ **ポインタが指すメモリの値を書き換える**
- ✔ ぬるぽ
- ✔ 適当なアドレスが入ったポインタ



* 間接参照演算子

ポインタの基本的な事

- ✔ ポインタを定義する
- ✔ ポインタに変数のアドレスを入れる
- ✔ ポインタが指すメモリの値を書き換える
- ✔ ぬるぽ
- ✔ 適当なアドレスが入ったポインタ

ポインタの基本的な事

- ✔ ポインタを定義する
- ✔ ポインタに変数のアドレスを入れる
- ✔ ポインタが指すメモリの値を書き換える
- ✔ ぬるぽ
- ✔ 適当なアドレスが入ったポインタ

変数のアドレスはどれ？

値	0	0	0	0	0	0	0	0	0
番地	10	11	12	13	14	15	16	17	18

- ✔ メモリのアドレスは1byteずつ割り振られている
- ✔ 1つの変数でメモリを1byte以上使うことは普通にある
- ✔ この時の変数のアドレスはどれなのか？

変数のアドレスはどれ？

値	0	0	0	0	0	0	0	0	0
番地	10	11	12	13	14	15	16	17	18

- ✔ メモリのアドレスは1byteずつ割り振られている
- ✔ 1つの変数でメモリを1byte以上使うことは普通にある
- ✔ この時の変数のアドレスはどれなのか？

変数のアドレスはどれ？

値	0	0	0	0	0	0	0	0	0
番地	10	11	12	13	14	15	16	17	18

- ✔ メモリのアドレスは1byteずつ割り振られている
- ✔ 1つの変数でメモリを1byte以上使うことは普通にある
- ✔ この時の変数のアドレスはどれなのか？

変数のアドレスはどれ？

値	0	0	0	0	0	0	0	0	0
番地	10	11	12	13	14	15	16	17	18

- ✔ メモリのアドレスは1byteずつ割り振られている
- ✔ 1つの変数でメモリを1byte以上使うことは普通にある
- ✔ この時の変数のアドレスはどれなのか？
- ✔ **変数のアドレス = 一番小さいアドレス**

ポインタのサイズは？

✔ 4種類のポインタを用意した、それぞれのサイズは？

```c
#include <stdio.h>

int main(void)
{
    char* p1;
    short* p2;
    int* p4;
    long long* p8;

    printf("p1のサイズは %d\n", sizeof(p1));
    printf("p2のサイズは %d\n", sizeof(p2));
    printf("p4のサイズは %d\n", sizeof(p4));
    printf("p8のサイズは %d\n", sizeof(p8));

    return 0;
}
```

値渡しとポインタ渡し

値渡し = 関数の引数に値を渡す

ポインタ渡し = 関数の引数にポインタ(アドレス)を渡す

値渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 引数やローカル変数は関数用に用意した別のメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		
0x1010		
0x1014		
0x1018	100	main関数の変数a

} main関数用のメモリ

値渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 関数呼び出しで指定した引数はその関数用に用意したメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		func1関数の引数x
0x1010		
0x1014		
0x1018	100	main関数の変数a

func1(a); // 関数実行時にメモリ確保

func1関数用のメモリ

main関数用のメモリ

値渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 関数呼び出しで指定した引数はその関数用に用意したメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C	100	func1関数の引数x
0x1010		
0x1014		
0x1018	100	main関数の変数a

func1関数用のメモリ

main関数用のメモリ

ポインタ渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 関数呼び出しで指定した引数はその関数用に用意したメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		
0x1010		
0x1014		
0x1018	100	main関数の変数a

} main関数用のメモリ

ポインタ渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 関数呼び出しで指定した引数はその関数用に用意したメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2(&a); // 関数実行時にメモリ確保

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し

✔ 一般的に関数を呼び出すとその関数で必要になるメモリが確保される

✔ 関数呼び出しで指定した引数はその関数用に用意したメモリ空間に渡される

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C	0x1018	func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し≒参照渡し

- ✔ **ポインタ渡しは他言語でいう参照渡しとほぼ同じモノ**
- ✔ 厳密にはちょっとした違いはある
- ✔ ポインタ渡し = ヌルポなど参照先がないアドレスでもなんでも渡せる
- ✔ 参照渡し = アドレス(参照先)に変数などがあるアドレスしか渡せない
- ✔ C言語に参照渡しはない、C++にはポインタ渡しと参照渡しがある

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し≒参照渡し

- ✔ ポインタ渡しは他言語でいう参照渡しとほぼ同じモノ
- ✔ 厳密にはちょっとした違いはある
- ✔ ポインタ渡し = ヌルポなど参照先がないアドレスでもなんでも渡せる
- ✔ 参照渡し = アドレス(参照先)に変数などがあるアドレスしか渡せない
- ✔ C言語に参照渡しはない、C++にはポインタ渡しと参照渡しがある

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し≒参照渡し

- ✔ ポインタ渡しは他言語でいう参照渡しとほぼ同じモノ
- ✔ 厳密にはちょっとした違いはある
- ✔ ポインタ渡し = ヌルポなど参照先がないアドレスでもなんでも渡せる
- ✔ 参照渡し = アドレス(参照先)に変数などがあるアドレスしか渡せない
- ✔ C言語に参照渡しはない、C++にはポインタ渡しと参照渡しがある

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C	0x0000	func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し≒参照渡し

- ✔ ポインタ渡しは他言語でいう参照渡しとほぼ同じモノ
- ✔ 厳密にはちょっとした違いはある
- ✔ ポインタ渡し = ヌルポなど参照先がないアドレスでもなんでも渡せる
- ✔ **参照渡し = アドレス(参照先)に変数などがあるアドレスしか渡せない**
- ✔ C言語に参照渡しはない、C++にはポインタ渡しと参照渡しがある

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C	0x1018	func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポインタ渡し≒参照渡し

- ✔ ポインタ渡しは他言語でいう参照渡しとほぼ同じモノ
- ✔ 厳密にはちょっとした違いはある
- ✔ ポインタ渡し = ヌルポなど参照先がないアドレスでもなんでも渡せる
- ✔ 参照渡し = アドレス(参照先)に変数などがあるアドレスしか渡せない
- ✔ C言語に参照渡しはない、C++にはポインタ渡しと参照渡しがある

アドレス	値	備考
0x1000		
0x1004		
0x1008		
0x100C		func2関数の引数p
0x1010		
0x1014		
0x1018	100	main関数の変数a

func2関数用のメモリ

main関数用のメモリ

ポイント

- ✔ ポインタかどうかというのは割とどうでもよかったりする
- ✔ 扱っているのがただの値なのか、アドレスなのかを意識するべし