



# ポインタとconst

プログラマのためのC言語 第17回

## ● 概要

- ✓ 変数とconst
- ✓ 変数とポインタとconst
- ✓ 配列とconst
- ✓ 配列とポインタとconst
- ✓ 考え方は他の言語でも基本同じ

- 知っているとこういうので悩まない(例:js)

```
const a = {  
  hoge: "hoge",  
  foo : "foo"  
};
```

```
a.hoge = "foo";
```



const付けたのに値変更できるやん!!!  
const意味ねえええ!!!

“

変数と`const`

## ● 普通の変数

**a**

値	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B

メモリ4byteを確保して0で初期化

`int a = 0;`

## ● 普通の変数

**a**

値	00	0A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B

`int a = 0;`

`a = 10; // 代入可能`

後からメモリの値を変更可能

- `const`を付けた変数 = 定数

**a**

値	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B

メモリ4byteを確保して0で初期化

`const int a = 0;`

- `const`を付けた変数 = 定数

**a**

値	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B



```
const int a = 0;
```

```
a = 10; // 代入不可
```

`const`を付けた変数は後から値を変えられない



“

変数とポインタと*const*

## ● 普通のポインタ

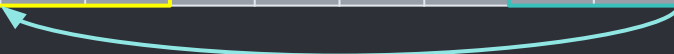
	a										p								
値	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	

```
const int a = 0;
```

```
int* p = &a;
```

## ● 普通のポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



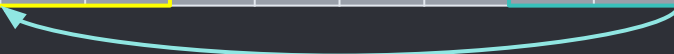
```
const int a = 0;
```

```
int* p = &a;
```

変数aのアドレスを入れる

## ● 普通のポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



```
const int a = 0;
```

```
int* p = &a;
```

```
*p = 10;
```

## ● 普通のポインタ

a						p													
値	00	0A	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	

```
const int a = 0;
```

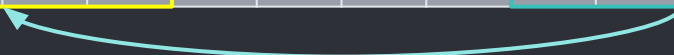
```
int* p = &a;
```

```
*p = 10;
```

大元の変数aにはconstがついているが  
ポインタ経由だったら普通に値変更できちゃう

## ● 普通のポインタ

	a					p													
値	00	0A	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	



```
const int a = 0;
```

```
int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

ポインタは参照先の値だけでなく  
当然自分の値も変更できる

## ● 普通のポインタ

	a										p								
値	00	0A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	

ヌルポ

```
const int a = 0;
```

```
int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

## ● 普通のポインタ

	a					p													
値	00	0A	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	

```
const int a = 0;
```

```
int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

ポインタは変更可能な対象が参照先と自分の2通りがある



## ● 普通のポインタ

	a					p													
値	00	0A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	

```
const int a = 0;
```

```
int* p = &a;
```

```
*p = 10;
```

ポインタは変更可能な対象が参照先と自分の2通りがある

```
p = 0;
```

つまり **const** の書き方も 2通りある

- 型の前に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
const int a = 0;
```

```
const int* p = &a;
```

型の前にconst

- 型の前に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
const int a = 0;
```

```
const int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

- 型の前に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



参照先は変更不可

```
const int a = 0;  
const int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

- 型の前に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



```
const int a = 0;  
const int* p = &a;
```

```
*p = 10;
```

```
p = 0;
```

ポインタの値は変更可能

- 型の後ろに **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
const int a = 0;
```

```
int* const p = &a;
```

型の後ろにconst

- 型の後ろに **const** を付けたポインタ

	a										p									
値	00	0A	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

参照先は変更可能

```
const int a = 0;
```

```
int* const p = &a;
```

```
*p = 10;
```

```
p = 0;
```

- 型の後ろに **const** を付けたポインタ

	a										p									
値	00	0A	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
const int a = 0;  
int* const p = &a;
```

```
*p = 10;
```

```
p = 0;
```



ポインタの値は変更不可



- 型の前後に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
const int a = 0;
```

```
const int* const p = &a;
```

型の前後にconst

- 型の前後に **const** を付けたポインタ

	a										p									
値	00	00	00	00	00	00	00	00	00	00	0B	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



`const int a = 0;`

`const int* const p = &a;`

`*p = 10;`

`p = 0;`



参照先、ポインタの値  
共に変更不可

## ● 中間まとめ

- ✓ ポインタは変更対象が参照先とポインタの値の2通りある
- ✓ よってconstの指定方法も2通りある

`const int* const p = &a;`

型の前のconstは参照先に作用

型の後ろのconstはポインタに作用

“

配列とconst

- constをつけた配列

```
const char a[] = {1, 2, 3, 4, 5};
```

charが5つで5byte

a

値	00	01	02	03	04	05	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B

- constをつけた配列

```
const char a[] = {1, 2, 3, 4, 5};
```

charが5つで5byte

a

値	00	01	02	03	04	05	00	00	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B



constついてるからダメ

```
a[0] = 0;
```

“

配列とポインタと*const*

- constを付けないポインタ

```
const char a[] = {1, 2, 3, 4, 5};
```

```
char* p = a;
```

	a						p													
値	00	01	02	03	04	05	00	00	00	0B	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
p[0] = 0;
```

```
p = 0;
```



- constを付けないポインタ

```
const char a[] = {1, 2, 3, 4, 5};
```

```
char* p = a;
```

	a						p													
値	00	01	02	03	04	05	00	00	00	0B	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
p[0] = 0;
```

```
p = 0;
```

参照先もポインタの値も変更可能

- 型の前にconstを付けたポインタ

```
const char a[] = {1, 2, 3, 4, 5};
```

```
const char* p = a;
```

	a						p												
値	00	01	02	03	04	05	00	00	00	0B	00	00	00	00	00	00	00	00	
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	



constついてるからダメ

```
p[0] = 0;
```

```
p = 0;
```

- 型の後ろにconstを付けたポインタ

```
const char a[] = {1, 2, 3, 4, 5};
```

```
char* const p = a;
```

	a						p													
値	00	01	02	03	04	05	00	00	00	0B	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		

```
p[0] = 0;
```

```
p = 0;
```



constついてるからダメ

- 型の前後にconstを付けたポインタ

```
const char a[] = {1, 2, 3, 4, 5};
```

```
const char* const p = a;
```

	a						p													
値	00	01	02	03	04	05	00	00	00	0B	00	00	00	00	00	00	00	00	00	00
番地	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B		



constついてるからダメ



constついてるからダメ

```
p[0] = 0;
```

```
p = 0;
```

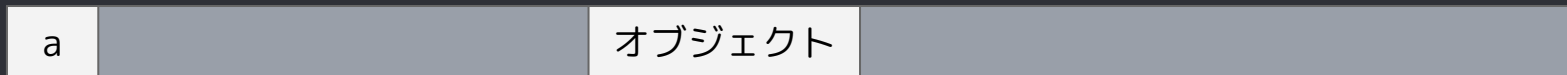
“

考え方は他の言語でも基本同じ

- 知っているとこういうので悩まない(例:js)

```
const a = {  
  hoge: "hoge",  
  foo : "foo"  
};
```

```
a.hoge = "foo";
```

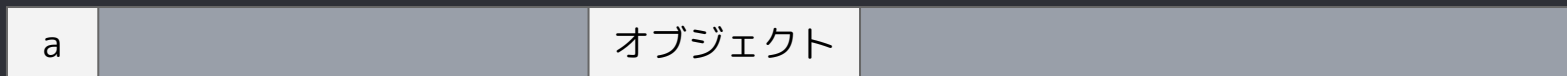


変数aはメモリ上のオブジェクトの場所  
つまりアドレスを覚えているだけ

- 知っているとこういうので悩まない(例:js)

```
const a = {  
  hoge: "hoge",  
  foo : "foo"  
};
```

```
a.hoge = "foo";
```



```
a = null;
```

a は constがついているのでこれは **NG**  
aの値は変えられない

- 知っているとなんかこのように悩まない(例:js)

```
const a = {  
  hoge: "hoge",  
  foo : "foo"  
};
```

```
a.hoge = "foo";
```

オブジェクトの値は変更可能

