「アルゴリズム」資料
2c. C のポインタと const

奈良女子大学理学部情報科学科
鴨浩靖

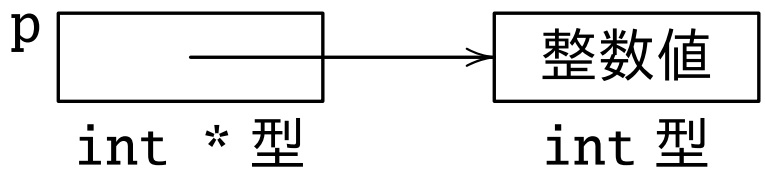
2011 年 10 月 24 日 初版
2012 年 12 月 20 日 第二版
2013 年 10 月 21 日 第三版
2020 年 10 月 12 日 第四版

C のポインタの復習

```
int     *p;     /* int へのポインタ型の変数 p を定義する  */
```

C の式で *p は p の指す実体を表すことから類推して、次のように考えるとわかりやすい。

p の実体をとると int になる。

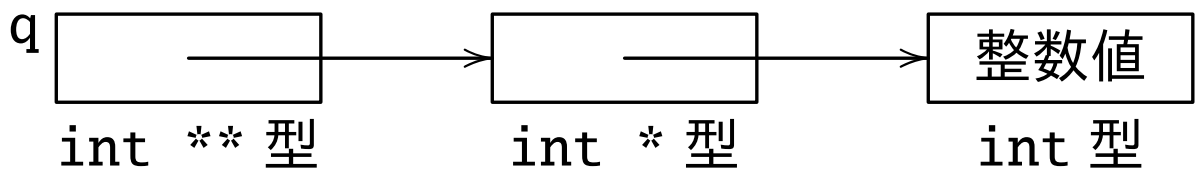


C のポインタの復習（つづき）

```
int     **q;
    /* int へのポインタへのポインタ型の変数 q を定義する*/
```

同じように、こう考えると良い。

p の実体をとって、さらに実体をとると、int になる。

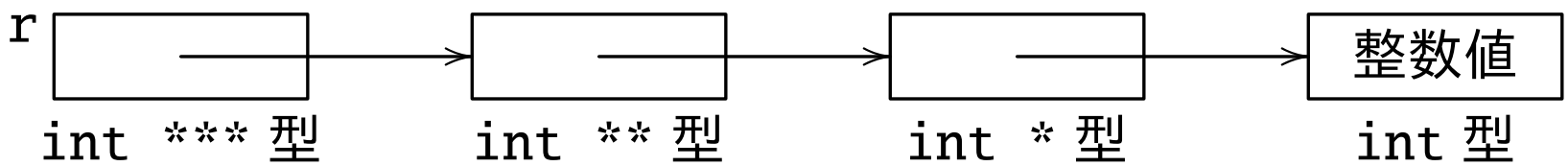


C のポインタの復習（つづきのつづき）

```
int      ***r;
    /* int へのポインタへのポインタへのポインタ型の変数 r を
定義する  */
```

これも同様。

p の実体をとって、さらに実体をとって、さらに実体をとると、int になる。

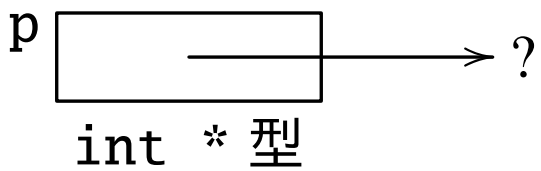


C のポインタの復習（補足）

ポインタ型の変数を定義しただけでは、実体は作られないことに十分に注意すること。

```
int     *p;     /* 初期化なし */
```

これだけでは、p にはゴミが入っていて、あらぬところを指すことになる。



const 型修飾子

変数定義で const をつけると、「値の変更不可」を意味する。変更しようとすると、通常はコンパイルエラーが発生する。

```
int         x = 3;
const int   y = 3;
```

と定義した場合、

可　　`x = 5;`

不可　`y = 5;`

const 型修飾子とポインタ

const とポインタの組み合わせでは、順序に注意。

```
int         x = 3, y = 2;
int         *p = &x;
const int   *q = &x;
int         *const r = &x;
const int   *const s = &x;
```

と定義した場合、

可	`p = &y;`	不可	`r = &y;`
可	`*p = 5;`	可	`*r = 5;`
可	`q = &y;`	不可	`q = &y;`
不可	`*q = 5;`	不可	`*q = 5;`

const 型修飾子とポインタ（つづき）

```
const int  *q = &x;     /* 実体が変更不可  */
```

q の実体をとると、int であり変更不可。

```
int  *const r = &x;     /* それ自体が変更不可  */
```

r は変更不可で、実体をとると int。

注意

const は、そのポインタを通して変更不可であることを要求するだけなので、他で変更されることはある。

例

```
int x = 3;
const int *p = &x;
use(*p);
x = 5;
use(*p);
```