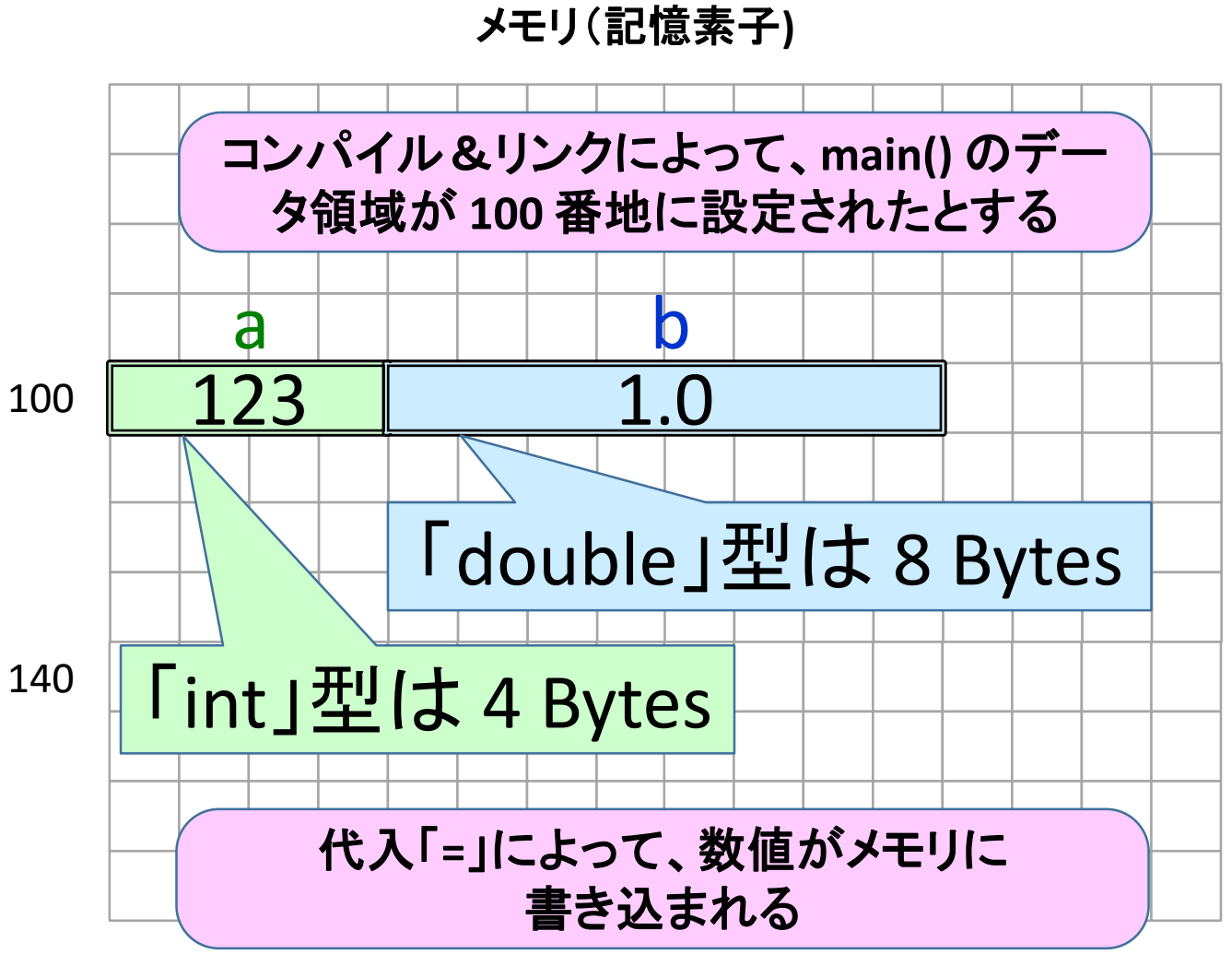
[B3]
ポインタ

Copyright © 2020-2024, by Takeshi Kawabata

変数宣言とメモリ領域

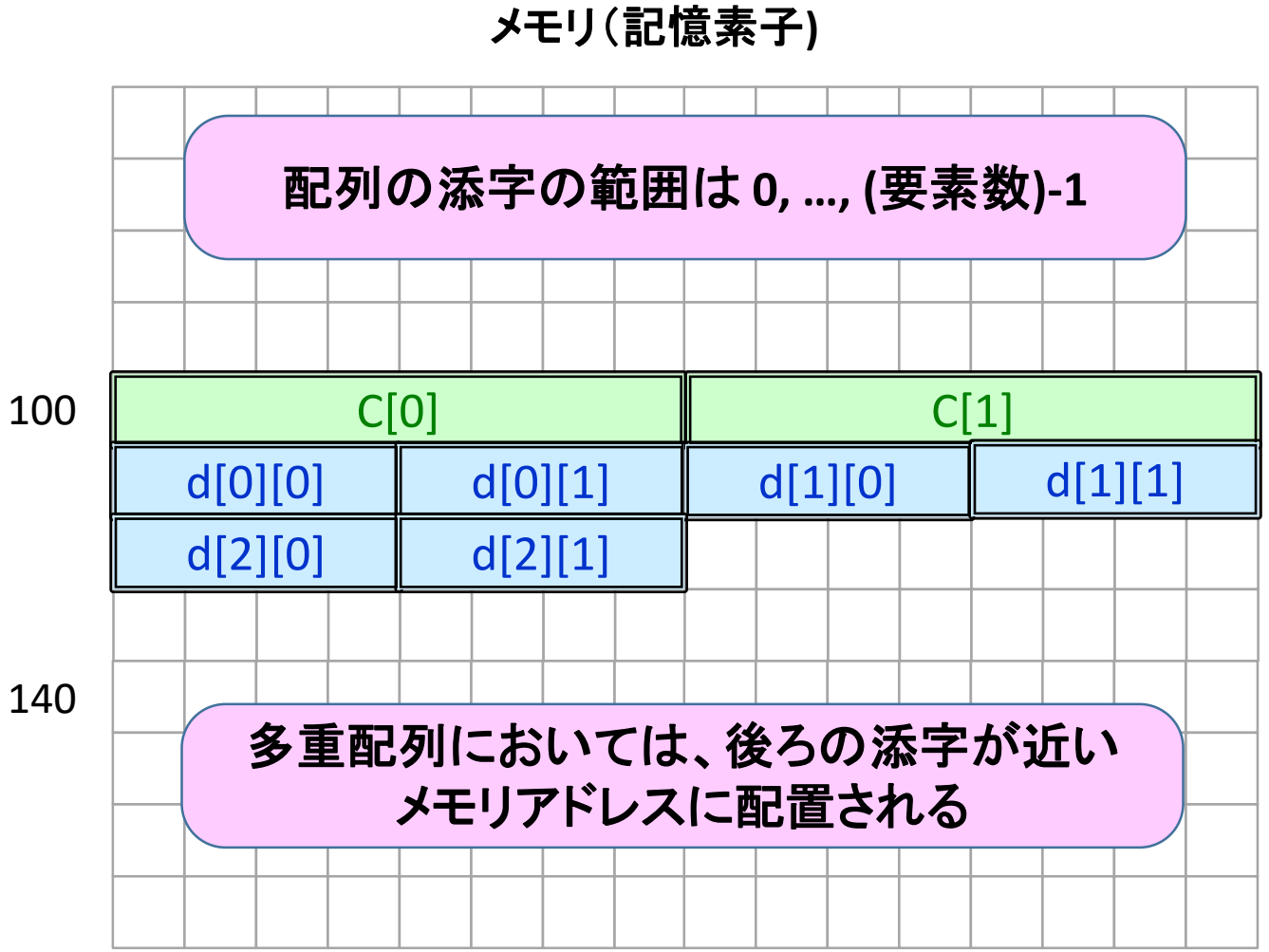
```
#include <stdio.h>
int main(void)
{
   int      a;
   double   b;

   a = 123;
   b = 1.0;

   return 0;
}
```

変数宣言によって、
メモリ領域を確保
a : 100 番地
b : 104 番地



配列宣言とメモリ領域

```
#include <stdio.h>
int main(void)
{
  double  c[2];
  int     d[3][2];
  …
  return 0;
}
```



ポインタ (pointer = 指し示すもの)

- ポインタ型
  - 「アドレス」を記憶するための変数の型
  - cygwin64 のポインタ型は 64 bits (8 Bytes)
- ポインタの宣言
  「(既存型名) ＊(ポインタ名);」

```
int     *pa; /* 整数変数へのポインタ          */
double  *pb; /* 浮動小数点変数へのポインタ  */
```

「&」演算子と「*」演算子

- 「&」演算子　(アンド、アンパサンド)
  - 既存の変数のアドレスを調べる

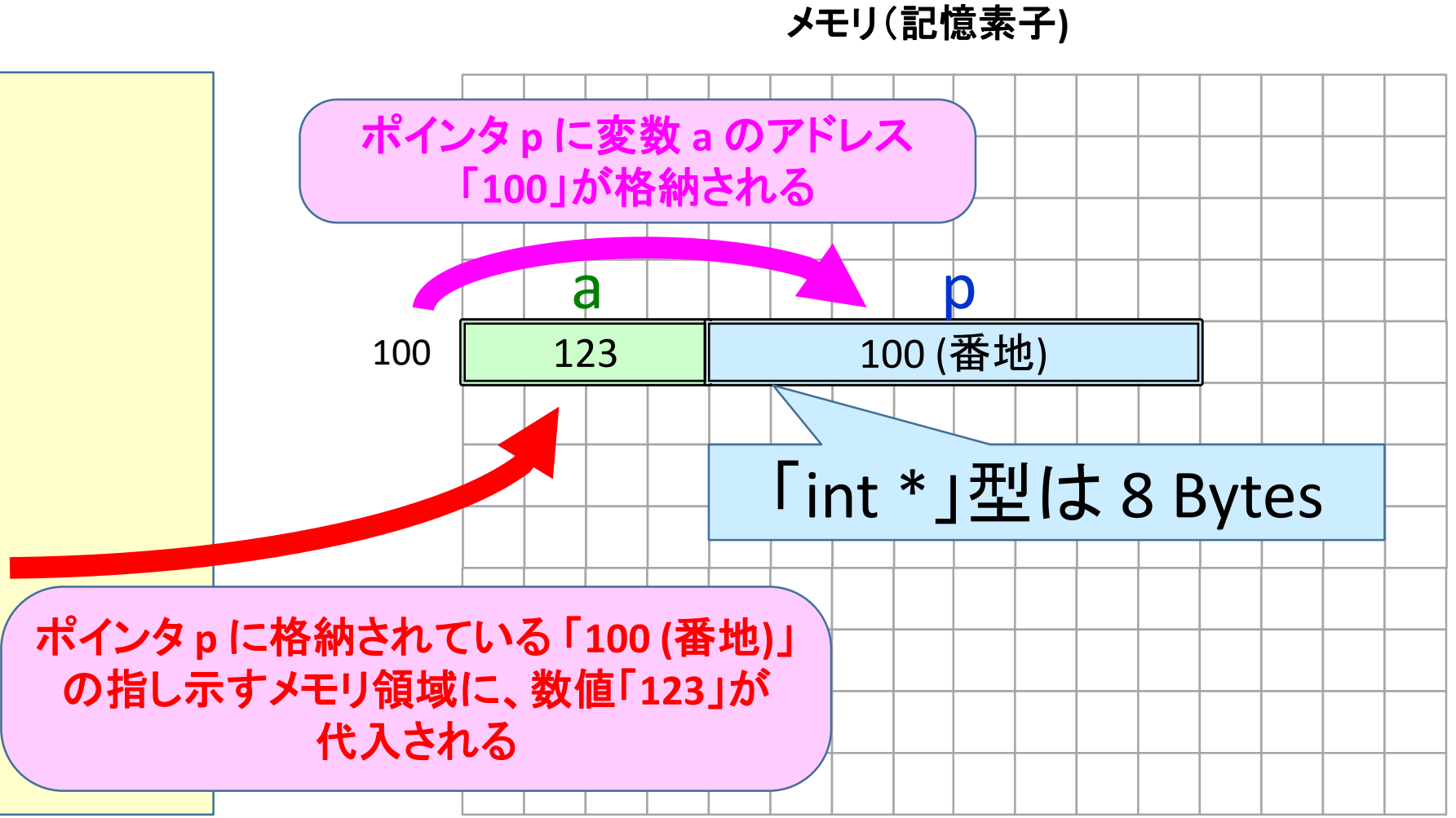
```
int  a, *p;
p = &a; /* 変数 a のアドレスをポインタ p に格納 */
```

- 「*」演算子　(星、スター、アスタリスク)
  - ポインタに格納されたアドレスのメモリ領域に対して読み書きを行う (間接アドレッシング)(宣言の*と違う)

```
*p = 123; /* 変数 a に数値 123 を代入 */
```

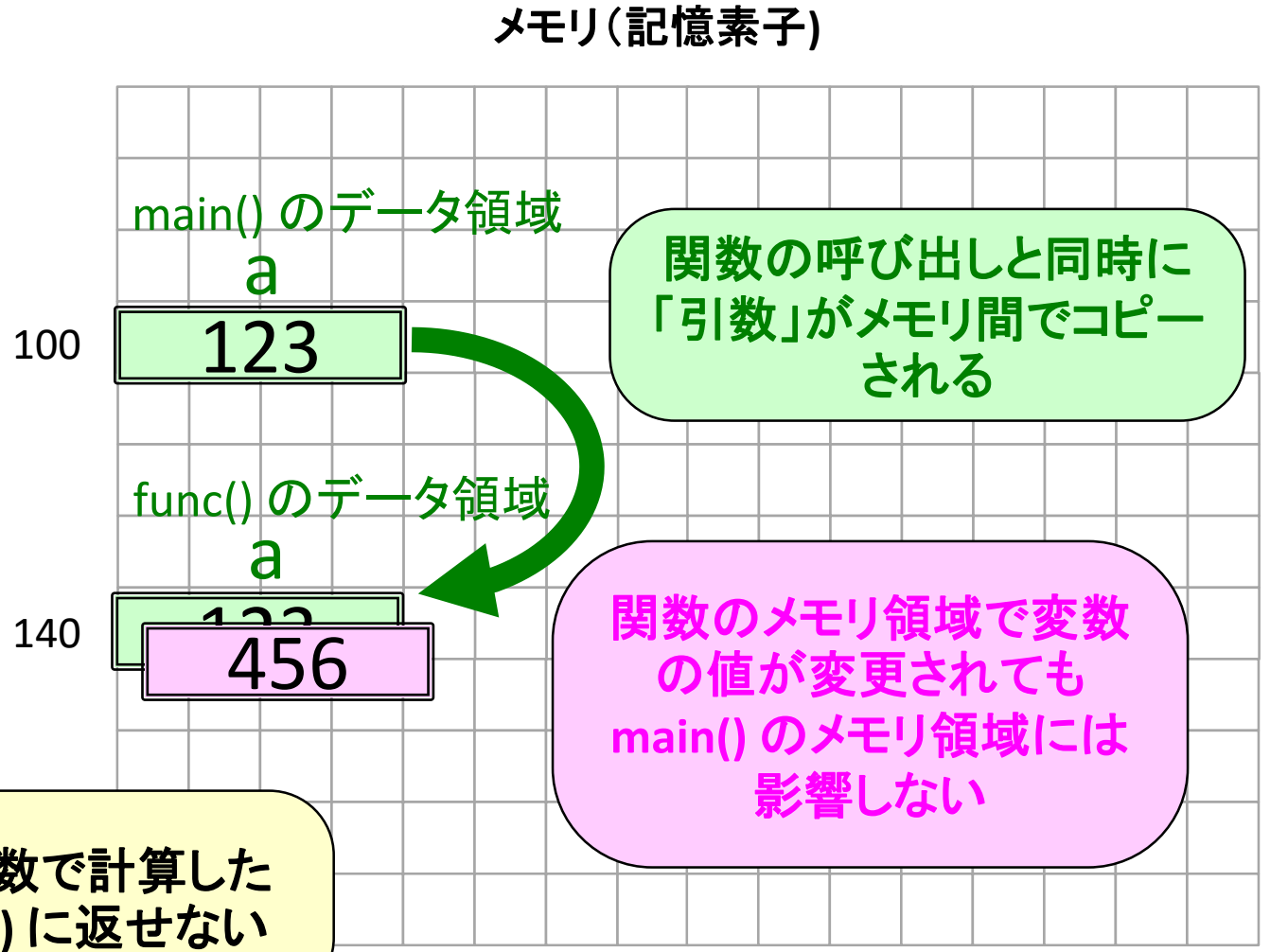
ポインタと「&」「*」演算子

```
#include <stdio.h>
int main(void)
{
   int      a, *p;

   p  = &a;

   *p = 123;

   return 0;
}
```



関数の呼び出しと引数のコピー

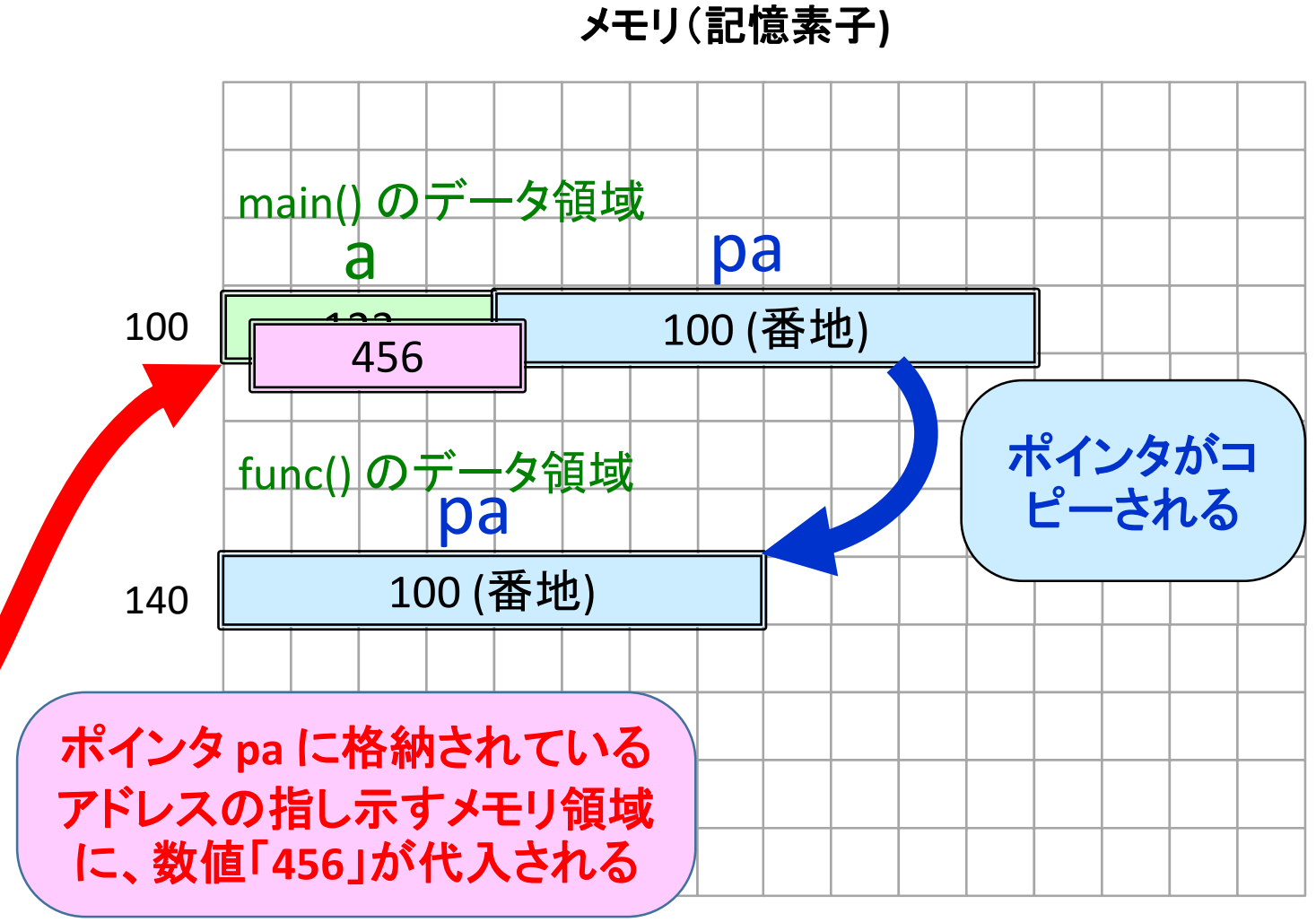
```
#include <stdio.h>
int main(void) {
   int  a=123;
   func(a);/* 引数は変数 */
   printf( “%d¥n” ,a);
   return 0;
}

void func(int a) {
   a = 456;
}
```



これでは、関数で計算した結果を main() に返せない

関数の呼び出しとポインタ

```
#include <stdio.h>
int main(void) {
   int a=123, *pa;
   pa = &a;
   func(pa);/* 引数はポインタ */
   printf( "%d¥n" ,a);
   return 0;
}

void func(int *pa) {
   *pa = 456;
}
```



構造体へのポインタ

```
#include <stdio.h>
#include <string.h>

typedef struct {
  char    name[32];
  int     price;
  double  weight;
} item_t;

int main(void) {
  item_t  a, *p;
  p = &a;
```

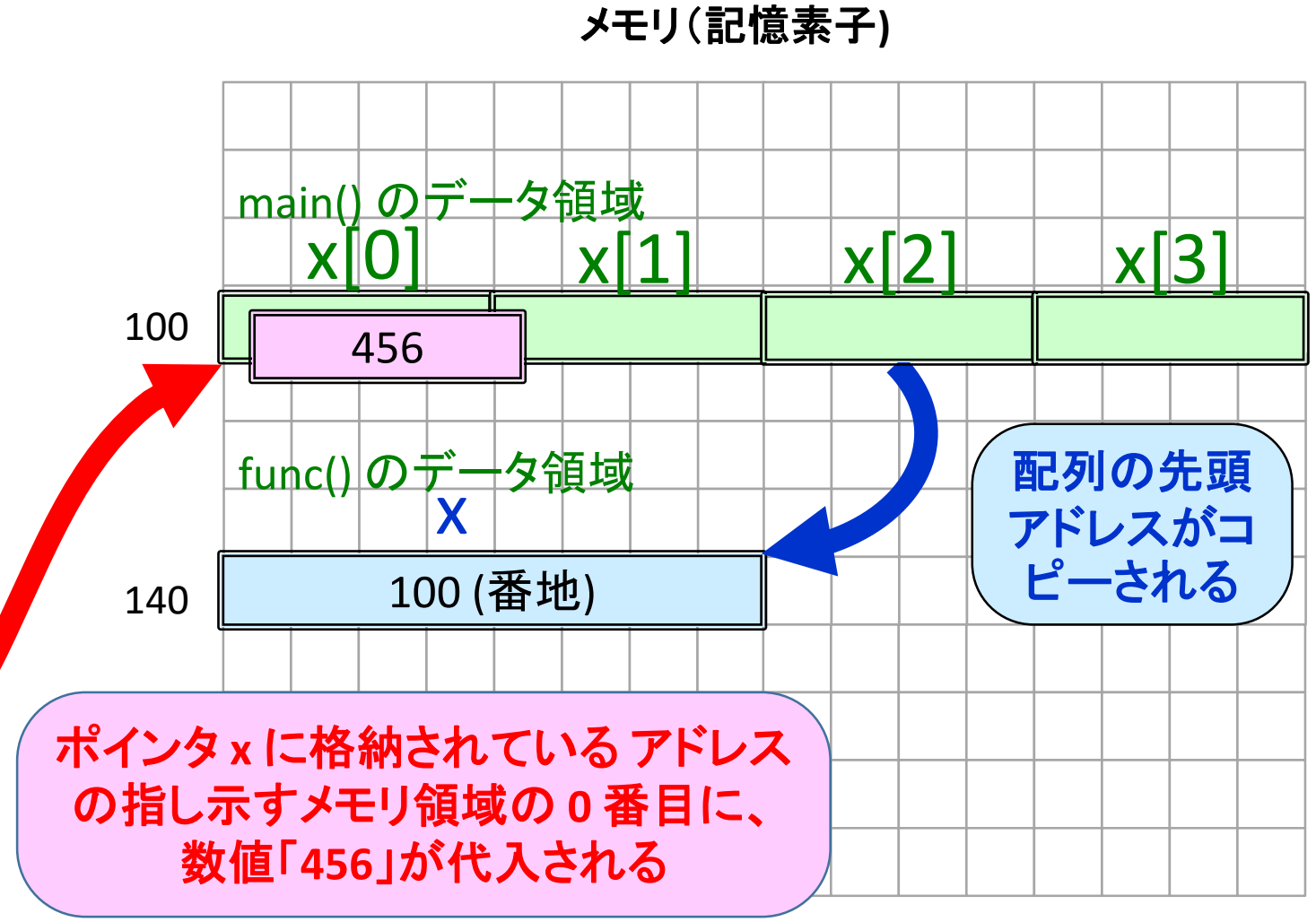
```
strncpy(a.name,  “apple”,31);
a.price  = 120;
a.weight = 205.8;

strncpy(p->name,  “apple”,31)
p->price  = 120;
p->weight = 205.8;
…
  return 0;
}
```

構造体へのポインタからメンバを指定するときは「.」でなく「->」を使用する

配列名はそのままポインタ

・配列名を単独で(後ろに「[ ]」を付けずに)使うと、
その配列のメモリ領域の先頭アドレスを表す

```
int x[10];

x[0] = 123;    /* ←同じ→ */      *x    = 123;

x[2] = 456;    /* ←同じ→ */      *(x+2) = 456;

printf( “%p¥n” , x);
```

配列名(アドレス)を引数として渡す

```
#include <stdio.h>
int main(void) {
   int x[4];
   func(x);/* 配列名そのまま */
   printf( “%d¥n” ,x[0]);
   return 0;
}

void func(int *x) {
   x[0] = 456;
}
```