

プログラミング基礎

<http://bit.ly/prog2d>


関数

前期 第2週

2017/4/17

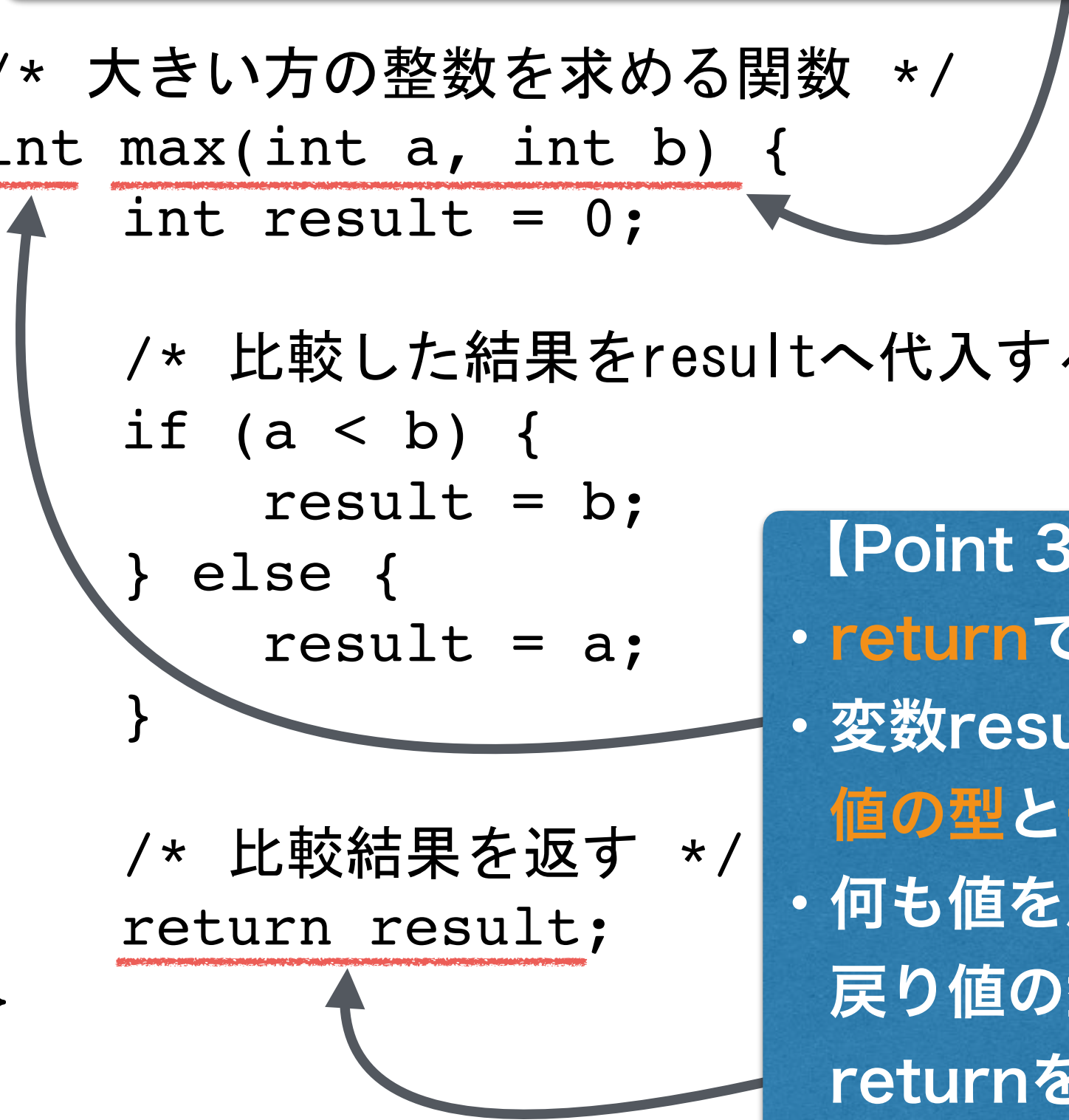
【Point 1】 関数の名前と引数などを先にプロトタイプ宣言しておく
(p. 261)

```
1: #include <stdio.h>
2:
3: /* 関数のプロトタイプ宣言 */
4: int max(int a, int b);
5:
```



**【Point 2】 戻り値の型 `max(仮引数1, 仮引数2, ...)`
の書式で書く (p. 218)**

```
6: /* 大きい方の整数を求める関数 */
7: int max(int a, int b) {
8:   int result = 0;
9:
10:  /* 比較した結果をresultへ代入する */
11:  if (a < b) {
12:    result = b;
13:  } else {
14:    result = a;
15:  }
16:
17:  /* 比較結果を返す */
18:  return result;
19: }
```



【Point 3】

- **return**で呼び出し元に値を返す
- 変数resultの型が、関数の**戻り値の型**と一致する必要がある
- 何も値を戻さない関数の場合、戻り値の型は**void**となり、**return**を省略できる (p. 236)

【Point 4】 仮引数aに5、仮引数bに10が代入される
(p. 226～p.227)

```
20:
21: int main(void)
22: {
23:     int x, y, z;
24:
25:     /* 実引数に値を直接書く場合 */
26:     z = max(5, 10);
```

```
27:     printf("max: %d\n", z);
28:
```

【Point 6】 関数max()の戻り値が
zに代入される (p. 239)

【Point 5】 関数実行後は、
returnの後にくる値に置き
換わる (この場合はresult代
入された10に置き換わる)
(p. 239 図8-11)


```
27: printf("max: %d\n", z);
```

```
28:
```

```
29: /* 実引数に変数を使う場合 */
```

```
30: printf("x > ");
```

```
31: scanf("%d", &x);
```

```
32: printf("y > ");
```

```
33: scanf("%d", &y);
```

```
34: z = max(x, y);
```

```
35: printf("max: %d\n", z);
```

```
36:
```

```
37: /* 戻り値を代入せずにそのまま使う場合 */
```

```
38: printf("max: %d\n", max(20, 50));
```

```
39: printf("max: %d\n", max(x, y));
```

```
40:
```

```
41: return 0;
```

```
42: }
```

【Point 7】 実引数には変数も使える

【Point 8】 戻り値は変数に代入することなく、呼び出しの記述をそのまま使うこともできる

【練習2-1】

サンプルプログラムをコンパイルして、
実行結果を確認しましょう。

【課題2-1】

関数max()を参考にして、引数で指定した2つの整数のうち小さい方を返す関数min()を作成してください。

[この関数のプロトタイプ宣言]

```
int min(int a, int b);  
/* max()の処理を参考に作ってみる */
```

[main()での処理]

```
printf("min: %d\n", min(5, 10));  
printf("min: %d\n", min(20, 10));
```

[実行例]

```
min: 5  
min: 10
```

【課題2-2】

3つの引数の値に対する平均値を求める関数
`average()`を作成してください。

[この関数のプロトタイプ宣言]

```
double average(double a, double b, double c);  
/* 3つの引数の平均値を求め、この関数の戻り値とする */
```

[`main()`での処理]

```
printf("average: %lf\n", average(30, 20, 40));  
printf("average: %lf\n", average(15, 8, 21));
```

[実行例]

```
average: 30.000000  
average: 14.666667
```


【課題2-3】

引数で指定した範囲内の整数の合計を求める関数 `sum_range()` を作成してください。

[この関数のプロトタイプ宣言]

```
int sum_range(int s, int e);
```

```
/* sからeの合計を求める */
```

```
/* s <= e となることを前提にしてよい */
```

```
/* (つまり s > e の場合を考慮しなくてもよい) */
```

【課題2-3】

[main()での処理]

```
printf("sum_range: %d\n", sum_range(3, 7));  
printf("sum_range: %d\n", sum_range(5, 10));
```

[実行例]

```
sum_range: 25  
sum_range: 45
```

【課題2-4】

2つの整数を指定した演算方法で計算する関数
`op_num()`を作成してください。

[この関数のプロトタイプ宣言]

```
int op_num(int a, int b, char op);
```

```
/* aとbを、opで指定した文字に応じて次のように計算し、*/
```

```
/* その結果を戻り値とする */
```

```
/*     • opが '+' の場合、 加算する */
```

```
/*     • opが '-' の場合、 減算する */
```

```
/*     • opが '*' の場合、 乗算する */
```

```
/*     • opが '/' の場合、 除算する（整数値として計算してよい） */
```

【課題2-4】

[main()での処理]

```
printf("op_num: %d\n", op_num(8, 3, '+'));  
printf("op_num: %d\n", op_num(6, 9, '*'));  
printf("op_num: %d\n", op_num(8, 3, '/'));  
printf("op_num: %d\n", op_num(2, 7, '-'));
```

[実行例]

```
op_num: 11  
op_num: 54  
op_num: 2  
op_num: -5
```

まだ余裕のある人は…

【課題2-5】

2つの整数 a , b の最大公約数を求める関数 $\text{gcd}()$ を作成してください。

最大公約数は、次のようなユークリッドの互除法を利用すると求めることができます。

- (1) a から b を割った余り t を求める
- (2) a に b を代入する
- (3) b に t を代入する
- (4) 上記の処理を、 b が0になるまで繰り返す (b は必ずいつか0なる)
- (5) 繰り返し終了時、つまり b が0となった時の a の値が最大公約数

(参考Webサイト)

<http://math-arithmetic.blogspot.jp/2011/01/blog-post.html>

【課題2-5】

[この関数のプロトタイプ宣言]

```
int gcd(int a, int b);
```

```
/* 繰り返し処理によって、前述のユークリッドの互除法を作る */
```

[main()での処理]

```
printf("gcd: %d\n", gcd(18, 12));  
printf("gcd: %d\n", gcd(221, 323));
```

[実行例]

```
gcd: 6  
gcd: 17
```