アルゴリズム
第6回授業
“引数と返却値”
(教科書 Page 34-42)

山口雅樹　（CISSP）

https://github.com/masakage/algorithm

本日の進め方

- 前回の復習（条件分岐）
- 引数と返却値
- 配列と繰り返し処理
- まとめ

1-7 引数と返却値（WIKIより）

　引数（ひきすう）は、コンピュータプログラムにおける手続きにおいて、その外部と値をやりとりするための特別な変数、あるいはその変数の値のことである。

　戻り値とは、プログラム中で呼び出された、関数、メソッド（クラス）、サブルーチンなどが処理を終了する際に、呼び出し元に対して渡す値の事である。

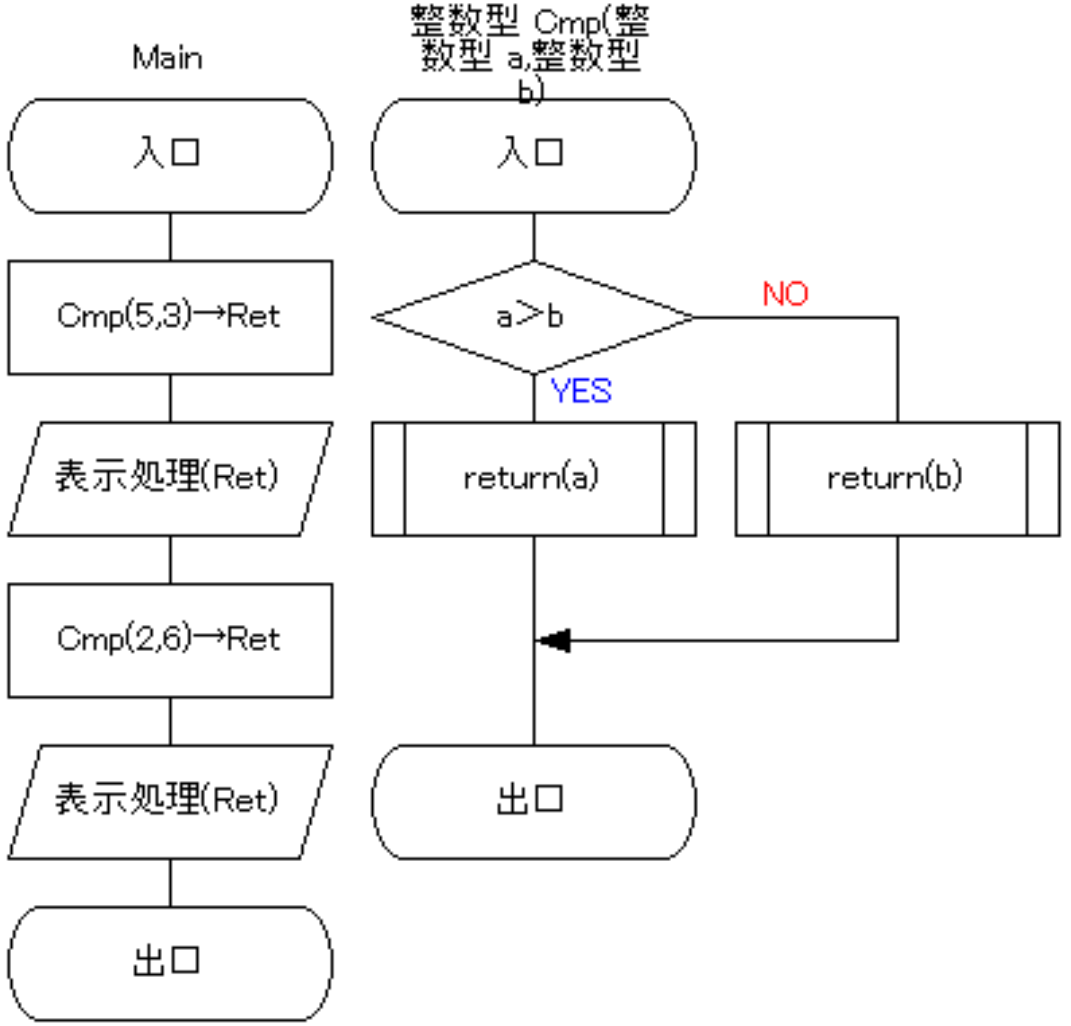
関数の呼び出しについて（引数と返却値）

```
○プログラム名：関数呼び出し /* 教科書 34ページサンプル */
○整数型：Ret

●Ret ← Cmp(5,3)
●表示処理(Ret)
●Ret ← Cmp(2,6)
●表示処理(Ret)
```

```
○整数型：Cmp(整数型：a,整数型：b)
▲a  >  b
|  ●return(a)
+ - - - - - - - - - -
|  ●return(b)
▼
```

関数

```
デバッグメッセージ / 出力:
実行..
------------------------------------------------------------
5
6
------------------------------------------------------------
```



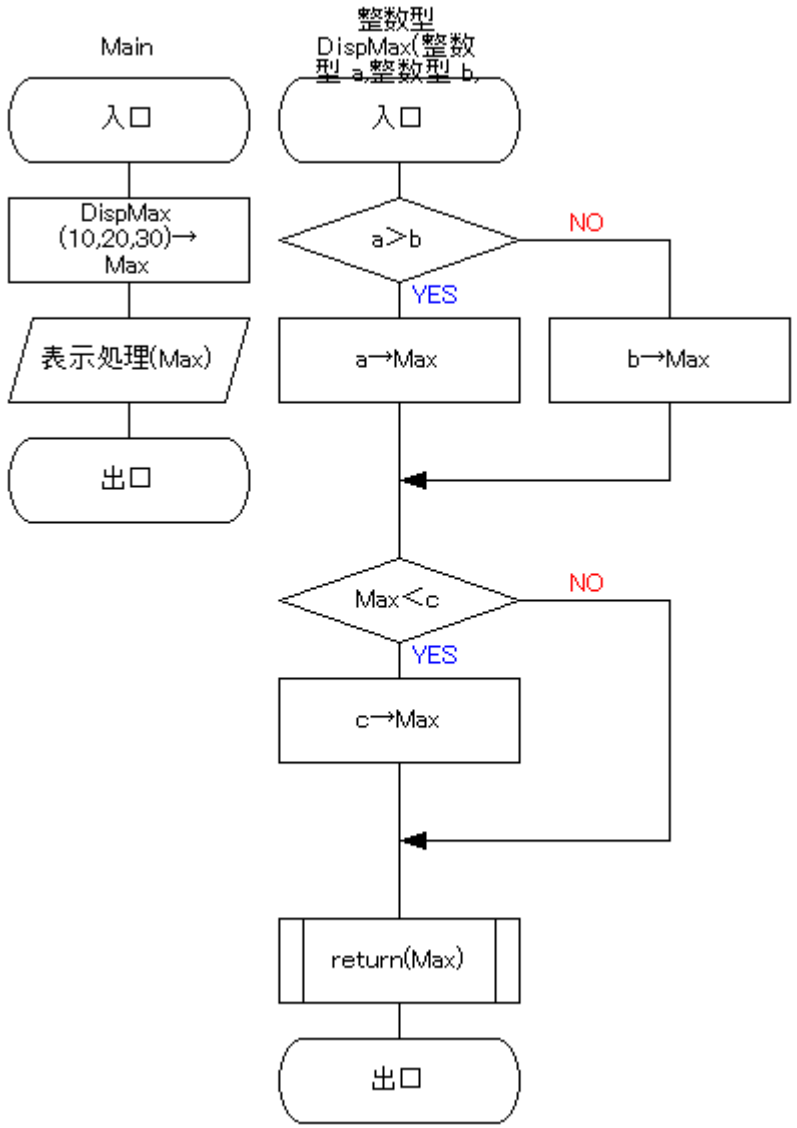
関数　（A,B,Cから最大値と求める）

```
○プログラム名：関数呼び出し /* 教科書 35ページサンプル */
○整数型：Max

●Max ← DispMax(10,20,30)
●表示処理(Max)
```

```
○整数型：DispMax(整数型：a,整数型：b,整数型：c)
○整数型：Max
▲a ＞ b
｜ ●Max ← a
＋－－－－－－－－－－
｜ ●Max ← b
▼

▲Max ＜ c
｜ ●Max ← c
▼

●return(Max)
```

関数

```
デバッグメッセージ / 出力:
実行..
----------------------------------------
30
```



関数　（入力値から求める）

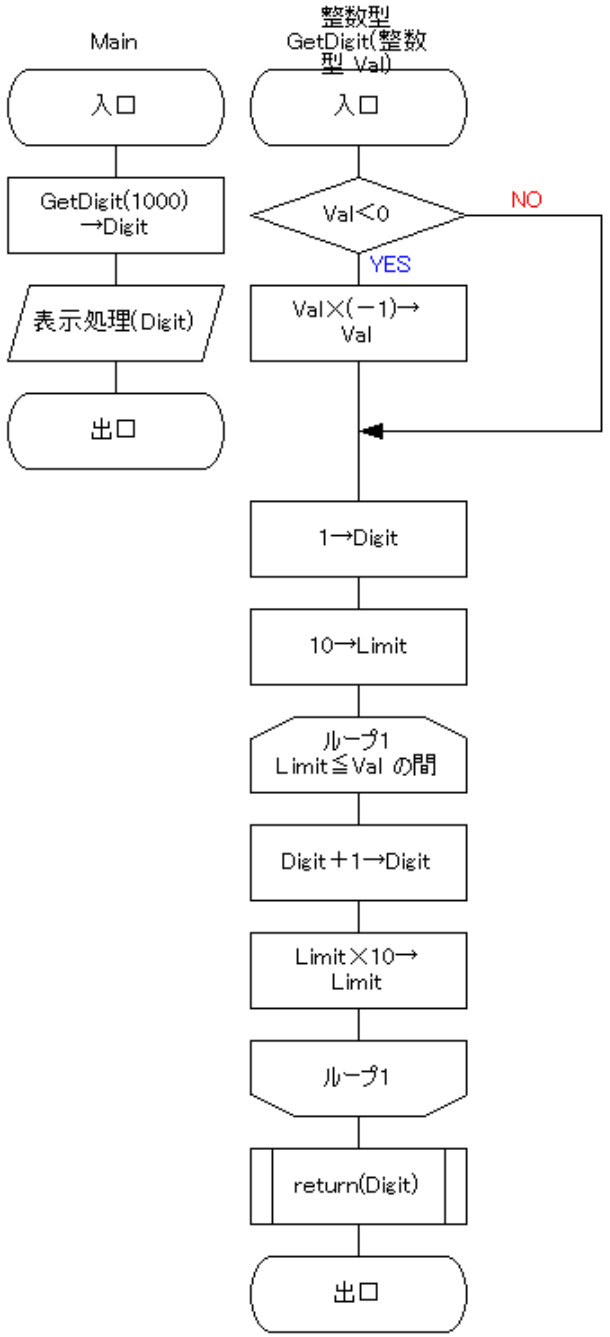
```
○プログラム名：関数呼び出し /* 教科書 35ページサンプル 桁数計算*/
○整数型：Digit

●Digit  ← GetDigit (1000)
●表示処理(Digit)


○整数型：GetDigit(整数型：Val)
○整数型：Digit,Limit
▲Val ＜ 0
|  ●Val ← Val × (-1)
▼

●Digit ← 1
●Limit ← 10

■Limit ≦ Val
|  ●Digit ← Digit + 1
|  ●Limit ← Limit × 10
□
●return(Digit)
```

関数

条件

繰り返し

```
デバッグメッセージ / 出力:
実行..
--------------------------------------------------------------------------------
4
```



1-8 配列と繰り返し処理（WIKIより）

　複数の要素（値）の集合を格納・管理するのに用いられるデータ構造が配列である。
1次元の配列は特に線形配列 (linear array) とも呼ばれる。
メモリ空間でもまとめて管理されているので効率がいい。
また、繰り返し処理を行いやすい。

　平均点を求める場合、ループを使いたい。。
しかし、配列がないと、生徒が増えたときに大変。。

1-8 配列と繰り返し処理

　配列の添え字　C#に合わせて　0　から始まります。
A[0] A[1] A[2] A[3] A[4] A[5] A[6] A[7] A[8] A[9]（合計10個）

　配列の宣言
　整数型：A[10]　（実際は、A[0]からはじまって、A[9]まで）

40ページのプログラムについて

```
○プログラム名：配列と繰り返し /* 教科書 40ページサンプル */
○整数型：A[10]

●A[0] ← 0
●A[1] ← 0
●A[2] ← 0
●A[3] ← 0
●A[4] ← 0
●A[5] ← 0
●A[6] ← 0
●A[7] ← 0
●A[8] ← 0
●A[9] ← 0

●表示処理(A[0])
●表示処理(A[1])
●表示処理(A[2])
●表示処理(A[3])
●表示処理(A[4])
●表示処理(A[5])
●表示処理(A[6])
●表示処理(A[7])
●表示処理(A[8])
●表示処理(A[9])
```

20行にもなる！

```
デバッグメッセージ / 出力:
実行..
----------------------------------------------------------------
0
0
0
0
0
0
0
0
0
0
```

```
○プログラム名：配列と繰り返し /* 教科書 40ページサンプル */
○整数型：A[10]
○整数型：Idx

■Idx：0,Idx ＜ 10,1
|   ●A[Idx] ← 0
|   ●表示処理(A[Idx])
□|
```

4行ですみます！

```
デバッグメッセージ / 出力:
実行..
----------------------------------------------------------------
0
0
0
0
0
0
0
0
0
0
```

平均点の算出　（教科書40Page Example）

```
○プログラム名：平均点 /* 教科書 40ページサンプル */
○整数型：Ten[5]
○整数型：Idx
○整数型：Gokei
○整数型：Heikin

●Ten[0] ← 36
●Ten[1] ← 80
●Ten[2] ← 100
●Ten[3] ← 92
●Ten[4] ← 64

●Gokei ← 0

■Idx：0,Idx ＜ 5,1
|   ●Gokei ← Gokei + Ten[Idx]
□

●Heikin ← Gokei ÷ 5
●表示処理 ("合計")
●表示処理(Gokei)
●表示処理 ("平均")
●表示処理(Heikin)
```

デバッグメッセージ / 出力:

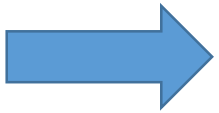
```
実行..
--------------------------------------------------
合計
372
平均
74
```

配列への代入（教科書41page Example)

○プログラム名：代入 /* 教科書 41ページサンプル */
○整数型：A[100]
○整数型：Idx

■Idx：0,Idx ＜ 100,1
｜　●A[Idx] ← Idx ＋ 1
□

■Idx：0,Idx ＜ 100,1
｜　●表示処理(A[Idx])
□



実行結果
1
2
3
4

省略

98
99
100

配列のコピー（教科書41page Example)

○プログラム名：配列コピー /* 教科書 41ページサンプル */
○整数型：From[10]
○整数型：To[10]

○整数型：Copy(整数型：From[])
○整数型：Idx
○整数型：To[10]
■Idx：0,Idx ＜ 10,1
｜　●To[Idx] ← From[Idx]
□
●return(To[])