



# C言語

菊地

再度詫び。





# お品書き

---

- 前回までのおさらい(paizaの立ち上げをば！)
- **条件文**の作り方！(if, if…else, if…else if文！  
+switch文)
- **簡易strlen**を作ろう！

前回まで  
のおさら  
い

- 第一回

- >出力関数**printf**, 繰り返し構文(**for**, **while**, **do...while**)

- 第二回

- >演算子、型、**printf**の使い方！

- 第三回

- >配列、演習問題

# 第一回

---

**printf**

---

**for文**

---

**While文**

---

**do…while文**

# 第一回

---

printf

←標準出力関数、  
第一引数を出力

---

for文 ←繰り返す回数がわかっている  
ときに使う

---

While文 ←繰り返す回数がわからない  
ときに使う

---

do…while文 ←1回以上繰り返したいと  
きに使う



# printfの使い方

- **定義**

**型**-> int型 :出力した文字数をバイト数で出力

**引数**->可変長引数。第一引数はconst char\*型

- **主な使い方**

**書式指定子**を第一引数の間に配置し、そこに対応した第二引数以降を表示する。

# 書式指定子

型に対応している。

書式指定子	対応する型	説明
<b>%c</b>	char	一文字を出力
<b>%s</b>	char *	文字列を出力
<b>%d</b>	int	整数を10進数で出力
<b>%u</b>	unsigned int	符号なし整数を10進数で出力
<b>%x</b>	unsigned int, int	整数を16進数で出力
<b>%f</b>	float, double	実数を出力



# やってみよう！

- paiza.ioというサイト (<https://paiza.io/ja>)で試しに書いてみよう！

3.これが出れば正解！



1. コード作成を試してみるを押す。



2. 左上の丸井ボタンの矢印を押してCを選択



# 復習問題！

- **Q1. 100**を19で割ったあまりの回数だけHi!って表示してみよう！

ヒント: 余りは余剰、つまり前回の演算子…%

(制限時間3分)



```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*whileのための変数*/
8      int i;
9
10     /*100を7で割ったあまり、つまり100%19=5*/
11     /*5回出力されるよ!*/
12     while(i < (100%19))
13     {
14         printf("Hi");
15         i++;
16     }
17     /*main関数の戻り値*/
18     return 0;
19 }
20
```

答え!

# 復習問題！

- **Q2.** 1.0 , 1.1 , 1.2, ... 2.0までを表示してみよう！

ヒント:繰り返し用の変数は処理の中でも使えるよ！たとえば、  
double i;

for(i = 1. ; i < 最大値; i += ?)

{printf(“%f”, ここにも使える ); ...}

おっとこれ以上は言えねえ…

(制限時間5分)

※小数点以下の0の数は気にしなくてok

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*for文のための変数*/
8      double i;
9
10     /*for文の書き方*/
11     /*for (初期値; 制限; 増減率)*/
12     /*今回は*/
13     /*for (1.0からスタート; 2.0まで; 0.1ずつ増える!)*/
14     /*i<2.1でも大丈夫!*/
15     for (i = 1.0; i <= 2.0; i += 0.1)
16     {
17         printf("%f\n", i);
18     }
19
20     /*main関数の戻り値*/
21     return 0;
```

答え！

## 第二回

### 間違えやすい演算子まとめ

演算子	意味
$x=y$	$x$ に $y$ を代入
$x==y$	$x$ と $y$ は等しいか？
$x>=y$	$x$ は $y$ 以上
$x(\text{算術演算子})=y$	$x(\text{算術演算子})y$ の結果を $x$ に代入

# インクリメント・デクリメント演算子

演算子	意味
$++x$	(前置演算)xに <b>1足してxを評価</b>
$x++$	(後置演算)xを <b>評価してからxに1足す</b>
$--x$	(前置演算)xから <b>1引いてxを評価</b>
$x--$	(後置演算)xを <b>評価してからxから1引く</b>

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      int i;
8
9      i = -3;
10
11     /*選択肢1*/
12     printf("Output 1\n");
13     while (i-- > 0)
14     {
15         printf("Hi?");
16     }
17     i = -3;
18
19     /*選択肢2*/
20     printf("\nOutput 2\n");
21     while (--i > 0)
22     {
23         printf("Hi!");
24     }
25 }
```

Q3.

以下の出力はどちらが多いかな？



# こたえ！

- 以下の出力はどちらが多いかな？
- 選択肢1のほうが多いよー！
- 3からはじまるのと、2から始まるのと

出力結果！

Output 1  
Hi?Hi?Hi?  
Output 2  
Hi!Hi!

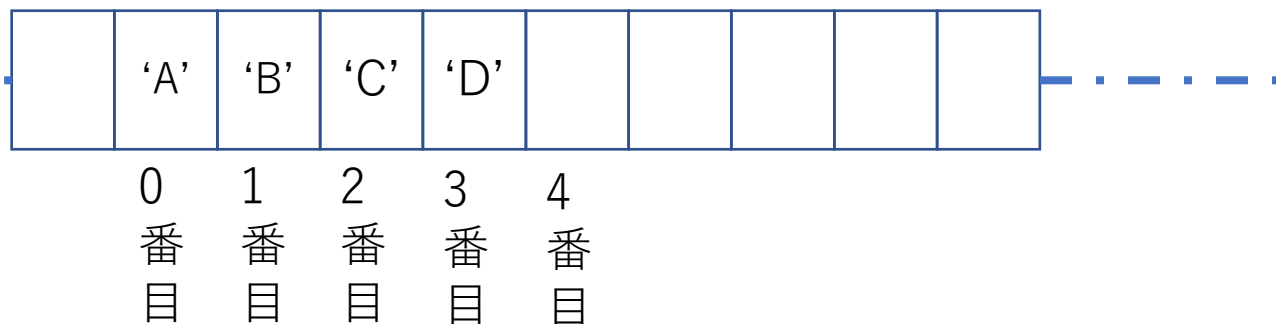
```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      int i;
8
9      i = -3;
10
11     /*選択肢1*/
12     printf("Output 1\n");
13     while (i-- > 0)
14     {
15         printf("Hi?");
16     }
17     i = -3;
18
19     /*選択肢2*/
20     printf("\nOutput 2\n");
21     while (--i > 0)
22     {
23         printf("Hi!");
24     }
25 }
```

# 第3回 配列

例えば

char Array[6] = "ABCD";  
と定義すると…

配列名  
=Array[5]



## ・定義

型名 配列名[配列の要素の数]

## ・要素の取り出し方

配列名[要素の先頭からの番号]

## ・注意点

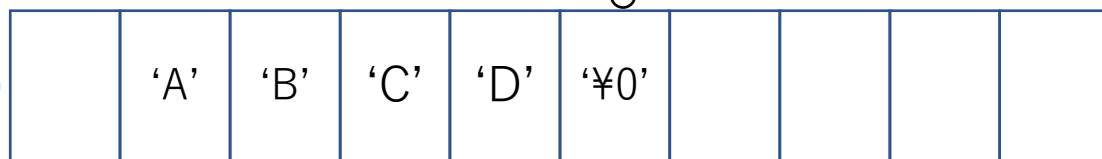
添え字の始まりは0から。最後は要素の最大値-1までしかない。

**初期値は不定！  
自分で入れないといけない！**

# 言い忘れていた！

実は終端文字('¥0'(null文字))が  
入ってる！

配列名  
=Array[5]



0  
番  
目

1  
番  
目

2  
番  
目


3  
番  
目

4  
番  
目

# ‘¥0’(null文字)とは

- かなり昔からの習わし…(紙テープの時はこれを送信している間に行頭に戻る時間を稼いで…いたとか…)
- c言語では**文字列の終端を表す重要な文字**。
- **値はintで見ると0!!!**
- 文章の途中にも挿入できる！





本題！

# 条件文について

- 指定された**条件を満たす場合**のみ、指定された**処理を行う**という構文。

- 例

**if**文(どこでも使える)

**if...else**文(どちらかに必ずあてはめたいとき)

**if...else if...**文(条件が複数ある場合)

**else, else ifはifがないと使えない！**



# if文

1. 条件文を書く
2. その下に条件を満たしたときに行う処理を書く。
3. 条件を満たすときのみ処理を行う。

```
1  /*参照するライブラリを指定*/  
2  #include <stdio.h>  
3  
4  /*戻り値の型 関数名 (引数の型)*/  
5  int main(void)  
6  {  
7      /*説明のための変数*/  
8      int i = 10;  
9  
10     /*1. if(条件(なんでも良子))*/  
11     if (i > 10)  
12     {  
13         /*2. 処理の内容*/  
14         printf("i is more than 10");  
15     }  
16  
17     /*main関数の戻り値*/  
18     return 0;  
19 }
```

# if...else文

1. if文を書く。
2. if文の条件に当てはまらない場合だけ、elseで指定された処理を行う。

elseには条件をかけない  
よ！！！！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*説明のための変数*/
8      int i = 10;
9
10     /*1. if(条件(なんでも良子))*/
11     if (i > 10)
12     {
13         /*1.5.処理の内容*/
14         printf("i is more than 10");
15     }
16     /*2. else*/
17     else
18     {
19         /*2.5処理の内容*/
20         printf("i is not more than 10");
21     }
22
23     /*main関数の戻り値*/
24     return 0;
```



# if...else if...文

1. if文を書く
2. ifで分岐させたくない条件等をelse ifの後ろに書く。
3. else ifの条件に当てはまる場合のみelse ifで指定された内容进行处理。

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*説明のための変数*/
8      int i = 7;
9
10     /*1. if(条件(なんでも良子))*/
11     if (i > 10)
12     {
13         /*1.5.処理の内容*/
14         printf("i is more than 10");
15     }
16     /*2. else if (条件文)*/
17     else if (i > 3 && i < 8)
18     {
19         /*2.5処理の内容*/
20         printf("i is more than 4 and i is less than 8");
21     }
22     /*3. else*/
23     else
24     {
25         /*3.5処理の内容*/
26         printf("i is not more than 10");
27     }
28
29     /*main関数の戻り値*/
30     return 0;
31 }
```

書いてみよう。

- Q1. 200を37で割った商が
- 5以上ならmore than 5
- 以下ならless than 5と表示してみよう！



# 解答例


```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*第一段階、値を求める*/
8      int i = 200 / 37;
9
10     /*解答例2*/
11     if (i >= 5)
12     {
13         /*処理の内容*/
14         printf("more than 5");
15     }
16     else if (i < 5)
17     {
18         /*処理の内容*/
19         printf("less than 5");
20     }
21
22     /*main関数の戻り値*/
23     return 0;
24 }
```

- Q1.
- 200を37で割った商が


5以上ならmore than 5、

以下ならless than 5と表示してみよう！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*第一段階、値を求める*/
8      int i = 200 / 37;
9
10     /*解答例1*/
11     if (i >= 5)
12     {
13         /*処理の内容*/
14         printf("more than 5");
15     }
16     else
17     {
18         /*処理の内容*/
19         printf("i is less than 5");
20     }
21
22     /*main関数の戻り値*/
23     return 0;
24 }
```



# 書いてみよう！

- Q2. forで10回処理を繰り返す間
  - 処理が1~3回目ならless than 3
  - 4~7回目ならmore than 4 and less than 7
  - それ以外はothersと出力してみよう！
- 

# 解答例

- Q2. forで10回処理を繰り返す間、
- 処理が1~3回目ならless than 3
- 4~7回目ならmore than 4 and less than 7
- それ以外はothersと出力してみよう！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  /*戻り値の型 関数名 (引数の型)*/
5  int main(void)
6  {
7      /*繰り返し用変数*/
8      int i;
9
10     /*繰り返し処理との組み合わせ!*/
11     /*もちろん、iが1からでもok。*/
12     /*その場合は分岐条件が異なるよ!*/
13     for (i = 0; i < 10; i++)
14     {
15         /*0からスタートなのに注意*/
16         /*1回目 = iは0*/
17         if (0 <= i && i <= 2)
18         {
19             printf("less than 3");
20         }
21         /*4~7回目、つまりiが3から6の時!*/
22         else if (3 <= i && i <= 6)
23         {
24             printf("more than 4 and less than 7");
25         }
26         /*そのほか!*/
27         else
28         {
29             printf("others");
30         }
31     }
32
33     /*main関数の戻り値*/
34     return 0;
35 }
```

# 簡易strlenを作ってみよう！

- strlenってなーに！  
(string.hにあるライブラリ関数)

## 今回作りたいもの！

## 定義

```
int my_strlen(char s[]);
```

## 引数

## char 型の配列

## 戻り値

## 文字列の大きさ

# テンプレート

これを参考に自分で考えてみよう！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  int my_strlen(char s[])
5  {
6      /*文字数を数えるための変数*/
7      int i = 0;
8
9      /*回数がわからない時の繰り返し構文*/
10     while (s[i] != '\0')
11     {
12         i++;
13     }
14     return (i);
15 }
16
17 /*戻り値の型 関数名 (引数の型)*/
18 int main(void)
19 {
20     /*確かめるための変数*/
21     char s[10] = "You!";
22
23     /*確認の仕方!*/
24     printf("s's length = %d", my_strlen(s));
25
26     /*main関数の戻り値*/
27     return 0;
28 }
```

# 答え

終端文字(null文字)まで数えればいいね！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  int my_strlen(char s[])
5  {
6      /*文字数を数えるための変数*/
7      int i = 0;
8
9      /*回数がわからない時の繰り返し構文*/
10     while (s[i] != '\0')
11     {
12         i++;
13     }
14     return (i);
15 }
16
17 /*戻り値の型 関数名 (引数の型)*/
18 int main(void)
19 {
20     /*確かめるための変数*/
21     char s[10] = "You!";
22
23     /*確認の仕方！*/
24     printf("s's length = %d", my_strlen(s));
25
26     /*main関数の戻り値*/
27     return 0;
28 }
```



# もう一回！

- strlenってなーに！  
(string.hにあるライブラリ関数)

今回作りたいもの！

## 定義

```
int my_strlen(char s[]);
```

## 引数

char 型の配列

## 戻り値

文字列の大きさ



# テンプレート

- my\_strlenの戻り値はint型の文字数
- 関数名はmy\_strlen
- 処理を書いてみよう！

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  int my_strlen(char s[])
5  {
6      /*ここに処理が来る！*/
7  }
8
9  /*戻り値の型 関数名 (引数の型)*/
10 int main(void)
11 {
12     /*確かめるための変数*/
13     char s[10] = "You!";
14
15     /*確認の仕方！*/
16     printf("s's length = %d", my_strlen(s));
17
18     /*main関数の戻り値*/
19     return 0;
20 }
```

# テンプレート2

- my\_strlenの戻り値はint型の文字数
- 関数名はmy\_strlen
- さあ、どこまで進めればいいのか？
- 構文は何を選べばいい？

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  int my_strlen(char s[])
5  {
6      /*文字数を数えるための変数*/
7      int i = 0;
8
9      /*回数がわからない時の繰り返し構文*/
10     OO()
11     {
12         /*iが終端に来るまで増やしてあげればいい!*/
13         i++;
14     }
15 }
16
17 /*戻り値の型 関数名 (引数の型)*/
18 int main(void)
19 {
20     /*確かめるための変数*/
21     char s[10] = "You!";
22
23     /*確認の仕方!*/
24     printf("s's length = %d", my_strlen(s));
25
26     /*main関数の戻り値*/
27     return 0;
28 }
```

# 答え

- 一人で書けるかな？

```
1  /*参照するライブラリを指定*/
2  #include <stdio.h>
3
4  int my_strlen(char s[])
5  {
6      /*文字数を数えるための変数*/
7      int i = 0;
8
9      /*回数がわからない時の繰り返し構文*/
10     while (s[i] != '\0')
11     {
12         i++;
13     }
14     return (i);
15 }
16
17 /*戻り値の型 関数名 (引数の型)*/
18 int main(void)
19 {
20     /*確かめるための変数*/
21     char s[10] = "You!";
22
23     /*確認の仕方!*/
24     printf("s's length = %d", my_strlen(s));
25
26     /*main関数の戻り値*/
27     return 0;
28 }
```



FIN