(・ω・)　(・ω・)

# 第3回　C言語講座！

## てーま：関数について(・ω・)!!

(・ω・)　(・ω・)

# まず・・・関数とは・・・？

☆今まで何も考えずに使ってきたmain関数やprintf関数、scanf関数…etc が関数と呼ばれているものです。

☆上記の他にも、文字列の長さを求める関数や…まぁいろいろな関数がありますが、

今回は、**自作関数**について簡単にお話します！

# 自作関数について(・ω・)!!

☆自作関数とは、読んで字のごとく関数を自分で自由に作ることです。　以上!!

☆詳しい作り方は次のページで→

# 関数の作り方

☆関数の作り方は…次の通りです！

```
返却値型　関数名(引き数){
        関数の内容
}
```

☆返却値型とは…main関数に値を返す時の値の型を指定しています。　後でまた説明します(・ω・)!!

☆関数名とは…関数の名前です。じぶんで自由に決めれます。

☆引き数とは…main関数から呼び出すときに持ってくる値の型を宣言しています。

# 関数の使用例①

```
#include<stdio.h>
int wa(int a, int b){
        int c;
        c=a+b;
        return c;
}
int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=wa(x, y);
        printf("z=%d¥n",z);
        return 0;
}
```

①

☆左は2つの値を入力して、その和を関数を使って求めるという簡単なプログラムです。

☆①のところで自作関数を使っています！

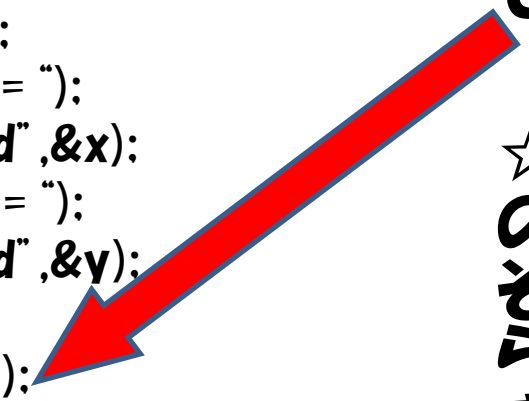
☆次のページで解説…(・ω・)

# 関数の使用例②

```
#include<stdio.h>
int wa(int a, int b){
        int c;
        c=a+b;
        return c;
}
int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=wa(x, y);

        printf("z=%d¥n",z);
        return 0;
}
```



☆まず自作関数はmain関数の前に書きます!!

☆ここでwa関数を呼び出しています。

☆main関数で入力したxとyの値がwa関数のaとbにそれぞれ入れられ、zにwa関数からの返却値cが代入されています。

つぎのページで例題やるよ!

# 例(・ω・)題①

2つの数を入力して、その値の差を関数を用いて求めよ。

☆ヒーント(・ω・)

3個前のスライドに関数の作り方は書いています。

☆またまたヒーント(・ω・)

1個前のスライドとほとんど変わりませんw

# 例題①こたえ

```
#include<stdio.h>
 int sa(int a, int b){
        int c;
        c=a-b;        ←←←←←ここの符号を変えるだけという簡単なものでしたが、
        return c;                                    関数の形を覚えれればいいと思う…(・ω・)
 }

 int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=sa(x, y);
        printf("z=%d¥n",z);
        return 0;
 }
```

# 補(・ω・)足

```
#include<stdio.h>
int sa(int a, int b){
        int c;
        c=a-b;
        return c;
}

int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=sa(x, y);
        printf("z=%d¥n",z);
        return 0;
}
```

☆関数内の処理が終わったら、その結果をmain関数に返す必要があります。これを返却値(戻り値)といいます。

☆左のは整数の計算だったので、返却値にはint型を使っている。なので返却値型はint型で宣言しています。

☆もし、小数を返すなら　double
☆もし、何も返さないなら　void(return不要)
と返却値型を宣言します

※返却値型をvoid型で宣言するときは、何も返さないのでreturn ○; は書きません。

# プロトタイプ宣言

新しい言葉が出てきましたが、たいしたことありません(・ω・)

ちょっとだけ書き方が変わるだけです。

今まではmain関数の前に関数を全部書いていましたが、これからはmain関数の前に関数名だけを宣言して、最後に関数の内容を書きます。

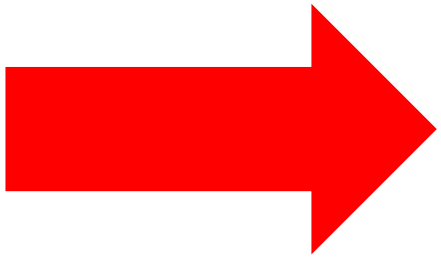
☆これが、**プロトタイプ宣言**です(・ω・)!!

# プロトタイプ宣言を使うと…

```
#include<stdio.h>
 int sa(int a, int b){
        int c;
        c=a-b;
        return c;
}

 int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=sa(x, y);
        printf("z=%d¥n",z);
        return 0;
}
```



```
#include<stdio.h>

int sa(int a, int b);  ←←プロトタイプ宣言
                          ※セミコロンを忘れずに!!
 int main(){
        int x, y, z;
        printf("x= ");
        scanf("%d",&x);
        printf("y= ");
        scanf("%d",&y);

        z=sa(x, y);
        printf("z=%d¥n",z);
        return 0;
}

int sa(int a, int b){    ←←あとから関数の
        int c;                    内容を書く
        c=a-b;
        return c;
}
```

簡単なので例題は省略します(・ω・)

# 再帰関数

再帰関数とは…

　関数内で自分自身の関数を呼び出すことを再帰関数と言います。

※注意※

終了する条件を設定しておかないと永遠に繰り返さるので気を付けましょう(・ω・`)

# 再帰関数を使った例

```
#include<stdio.h>
void cd(int);
void main(){
        int n;
        printf("自然数を入力してください:");
        scanf("%d",&n);

       cd(n);
}
void cd(int n){
        if(n>=0){
                printf("%d¥n",n);
                cd(n-1);
        }
}
```

☆このプログラムは自然数を入力すると0になるまで繰り返し1ずつ値を減らし出力するものです。

☆ここの条件がないと無限ループします。

☆左のを写して実行してみましょう(・ω・)!!

☆次のページは例題です。今日さいごの。

# 例(・ω・)題②

任意の自然数nを入力して、そのnの階乗を再帰関数を用いて求めましょう。

☆ヒーント

n＝0ならば 0! ＝ 1

n≠0 ならば n! ＝ n ＊ (n － 1)!

ifを使って分岐させましょう(・ω・)

# 例題②こたえ

```
#include<stdio.h>
int factorial(int n);

int main(){
        int n, result;
        printf("nを入力してください:");
        scanf("%d",&n);
        result=factorial(n);
        printf("%dの階乗は%dです(・ω・)!!¥n",n, result);
        return 0;
}
int factorial(int n){
        if(n==0){
                return 1;
        }
        else{
                return n*factorial(n-1);
        }
}
```

(・ω・)　(・ω・)

# お(・ω・)わ(・ω・)り
# 次も頑張ってね!!

(・ω・)　(・ω・)