C言語講座

第1回

第1回目の内容

入出力（printf,scanf)

四則演算

int型char型

演算子

分岐（if,switch）

ループ（while,for）

配列

新しいプロジェクトの作り方♪

1.Microsoft Visual studio 2010C++起動

2.「ツール」→「設定」→「上級者用の設定」を選択

3.「新しいプロジェクト」→「win32コンソールアプリケーション」を選択し、名前をここでは「lesson1」とします。

4. アプリケーションウィザードが出るので、「次へ」で「空のアプリケーション」を選択、「完了」

新規ソースファイルの生成

1. 「ソースファイル」のフォルダを右クリック。
2. 「追加」→「新しい項目」を選択。
3. 「C＋＋ファイル」を選択、名前を付けます。

（日本語以外で）、「追加」を選択。

とりあえず書いてみよう！！

```
#include <stdio.h>
int main(){
   printf("Hello World¥n");
   return 0;
}
```

「デバック」→「デバックなしで開始」で実行（Ctrl＋F5）

ソースの説明

#include <stdio.h>やint main()については後で説明します。

printf ・・・コンソール画面に標準出力するものです。

printf(“ここの文字を出力”)

拡張表記

“￥n”のことを拡張表記といいます。

拡張表記とは？・・・プログラミング言語などで、特別な文字列を表す表記です。また、文字定数とも呼ばれます。

printf(“Hello World¥n”)

¥n	改行
¥b	バックスペース
¥b	タブ
¥a	警報

変数と変換指定子

```
#include <stdio.h>


int main(){
   int a;
   a=100;
   printf("a=%d",a);
   return 0;
}
```

変数

変数・・・数値などを入れて置くための箱です。

int　a; **←このように変数を事前に宣言します。**

基本形	型	範囲
Int	整数型	-2,147,483,648～2,147,483,647
short	整数型	-32,768～32,767
long	整数型	-9,223,372,036,854,775,808～ 9,223,372,036,854,775,807
char	文字型	1文字
float	浮動小数型	-1.79769313486231570*10^308～ 1.79769313486231570*10^308
double	浮動小数型	-1.175494*10^38～1.175494*10^38

変換指定子

変換指定子とは？・・・“%d”のようなやつの“d”のこと

文字以外のものを文字に変換する機能を持っていて、“%”に続く文字によってかわります。

%d	整数を10進数で表示
%o	整数を8進数で表示
%x	整数を16進数で表示
%lf	実数を小数点につき10進数で表示
%c	引数に対応した1文字を表示
%5d	整数を10進数、5桁で表示。 開いているとこは空白　数字はいくつでもよい
%05d	整数を10進数、5桁で表示、開いているとこは0 数字はいくつでもよい

演算子

加減演算子

a+b	aとbの和
a-b	aとbの差

乗除演算子

a*b	aとbの積
a/b	aをbで割った商（整数どうしの場合、少数点以下は切り捨て）
a%b	aをbで割った余り（aとbは整数でないといけない）

変数と変換指定子

```
#include<stdio.h>

int main(){
    int a,b,c;
       a=100;
       b=90;
       c=a+b;
      printf("%d+%d=%dです。¥n"a,b,c);
return 0;
}
```

入力

```
#include<stdio.h>


void main(){
  int a;
   printf("整数を入力してください⇒");
   scanf("%d",&a);
   printf("入力した数字は%dです。¥n",a);


}
```

入力のソース解説

scanf・・・キーボードから数値などを読み込むために用いる関数

注）“&”を付けること。以下のようにつけて

scanf(“%d”,&a);

どうして＆を付けるのかについては、ポインタを学ぶときに教えます。
今回はおまじない程度でいいので覚えていてください。

CHARACTER型

```
#include<stdio.h>


void main(){
    char a;
     printf("一文字入力してください⇒");
     scanf("%c",&a);
     printf("入力された文字は「%c」です。¥n",a);


  }
```

CHARACTER型

char型・・・文字を記憶するための変数。

```
char b＝’b’;
```

注）文字を表すためには‘’を使います。char型は１文字しか表せません。

演習1

もしもし。

こんにちは。

それでは。

と表示するプログラムを生成してください。
注）printfは1回しか使えません。

演習2

2つの変数を用意して、それらに、scanfで10進数を入力してから、

a+b,　a-b,　a*b,　a/b,　a%b,

を画面に出力してください。

（ただし、bが0であることは考慮しなくてもいい。また、‘%’を出力するときは、%%と書く）

1の解答例

```
#include<stdio.h>

void main(){
        printf("もしもし。¥nこんにちは。¥n¥nそれでは。");
}
```

2の解答例

```
#include<stdio.h>

 void main(){
              int a,b;
             printf("a=");scanf("%d",&a);
             printf("b=");scanf("%d",&b);
             printf("a+b=%d¥n",a+b);
             printf("a-b=%d¥n",a-b);
             printf("a*b=%d¥n",a*b);
             printf("a/b=%d¥n",a/b);
             printf("a%%b=%d¥n",a%b);
}
```

演算子

・比較演算子

＝＝(等しい)

＝！(等しくない)

注)‘＝’は代入演算子なので、＝＝と区別しましょう

＜、＜＝、＞＝、＞ (大小関係)

・論理演算子

！　　（NOT：否定）

＆＆　（AND：かつ）

｜｜　（OR：または）

演算子

正しい演算のときは真（1：True）

異なる演算のときは偽（0:False）を返します。

分岐処理Ⅰ（IF文）

if・・・条件式によって処理を変える

```
if(条件式1){
   処理1;
}//条件式1が真のとき、処理1を行う

else if(条件式2){
   処理2;
}//条件式1が偽で、条件式2が真のとき、処理2を行う

else{
処理3;
}//どの処理も行わなければ、処理3を行う
```

例文

```
#include<stdio.h>

void main(){
  int num;
       printf("整数を入力してください");
   scanf("%d",&num);
  if(num%5==0){
          printf("5で割り切れます¥n");
   }else{
        printf("5で割り切れない¥n");
    }
}
```

分岐処理Ⅱ(SWITCH文)

switch文・・・条件式によって処理を変える

※switch文は条件判断を繰り返さないので、処理はif文よりも速い。

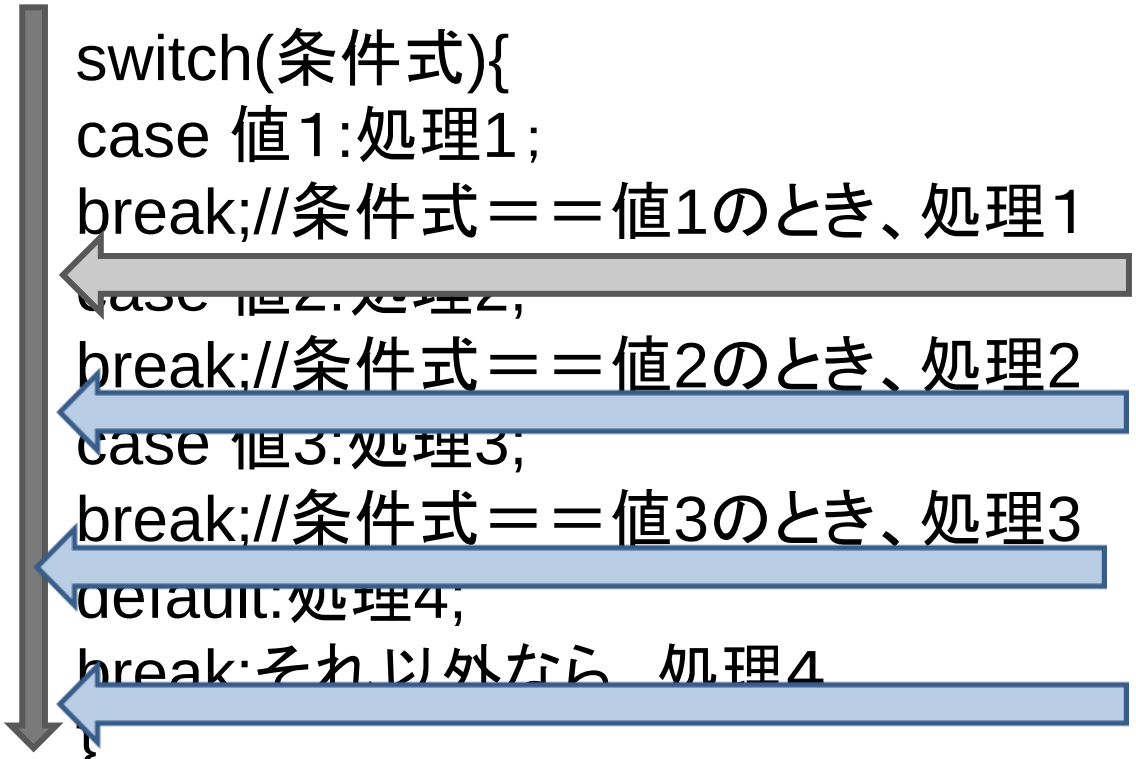
```
switch(条件式){
case 値1:処理1;
break;//条件式＝＝値1のとき、処理1
case 値2:処理2;
break;//条件式＝＝値2のとき、処理2
case 値3:処理3;
break;//条件式＝＝値3のとき、処理3
default:処理4;
break;それ以外なら、処理4
}
```

例文

```
#include<stdio.h>
void main(){
        int in;
         printf("数字をいれてください¥n");
        scanf("%d",&in);
 switch(in){
  case 1:
        printf("1が入力されました。¥n");
        break;
 case 2:
         printf("2が入力されました。¥n");
        break;
 default:
         printf("1、2以外の数字が入力されました。¥n");
        break;
 }
}
```

注）処理の後には必ず、break;をつける

これがないと、意図しない処理まで、行われる。



繰り返し文

同じような処理を繰り返すことができる。

do-while文、while文、for文が繰り返し文にはある。

```
do-while文
do{
処理;//条件式を満たす間、繰り返す
}while(条件式);

while文
while(条件式){
処理;//条件式を満たす間、繰り返す
}
```

DO-WHILE文とWHILE文の違い

```
#include<stdio.h>

void main(){
        int num=0;
        do{
                printf("do-while");
        }while(num==1);
}
```

do-while文

```
#include<stdio.h>

void main(){
        int num=0;
        while(num==1){
                printf("while¥n");
        }
}
```

while文

do-while文はループ内の処理の後で、条件式の判定を行う
while文はループ内の処理の前で、条件式の判定を行う

FOR文

繰り返しの処理の回数を指定することができる。

for文

for(初期化式;条件式;カウンタ){

処理;//条件を満たす間、繰り返す。

}

例文

```
#include<stdio.h>


void main(){
        int sum=0;
        int i;
        for(i=1;i<=5;i++){
                sum+=i;
        }
        printf("%d¥n",sum);
}
```

BREAK文

ループ処理を抜け出せる。

抜け出せるループはbreak1つにつき1つ

例文

```
#include<stdio.h>

void main(){
        int num=0;
        while(1){
                if(num>10){break;}
                        printf("%d",num);
                        num+=1;
        }
}
```

CONTINUE文

breakは完全にループを抜けるが、continueは1時的に処理を抜けることができる。

例文

```
#include<stdio.h>

void main(){
        int num=0;
        while(num<9){
                num+=1;
                if(num%2==1){continue;}
                        printf("%d",num);
        }
}
```

配列

同じデータ型を格納する変数をまとめて、管理することができる

要素の番号は0から始まる

構文　データ型　配列名[要素数];

例：int a[2]・・・数字を入れる

　　char s[5]・・・文字をいれる

多次元配列

データ型　配列名[横の要素数][縦の要素数]

例文

```
int a[5][5];
    a[2][3]=3;
    a[3][4]=1;
```

練習問題

評価A+,A,B,C,D,Eのいずれかを入力すると、その成績評価をとるために必要な基準得点の割合が表示されるプログラムを生成してください。

入力は5回できるようにしてください、また、評価の入力には配列を使ってください。

評価基準

A+・・・100~90

A・・・89~80

B・・・79~70

C・・・69~60

D・・・59~0

ヒント

If文

判断には‘¥0’をつかいます。

¥0・・・文字列の終わりの意味

→評価A・・・s[0]がA、s[1]が¥0