ソフトゼミC
第1回
C言語の復習

2013/08/05

エレクトロニクス研究部

今日は何をするの？

- printf/scanf
- If文/for文/while文
- 配列
- 構造体/関数
- ポインタ

→一気に総復習します！

printf -標準出力- (1)

- 最も初歩的な形式

printf(“Hello world!¥n”);

→Hello world!

- 特殊文字(エスケープシーケンス)

¥n →改行

¥a →警報音

文字として”‘ ¥を出力するときも前に¥を付ける

例　printf(“¥¥3000¥n”);

→¥3000

printf -標準出力- (2)

- 変数の出力

int a = 100;

printf(“a=%d¥n”,a);

→a=100

- 変換指定文字列(%d)の一覧

変換指定文字	使う型
%d	int型
%f	double型
%s	char型(文字列として出力)
%c	char型(1文字だけ出力)

scanf –標準入力-

- キーボードから文字を入力

scanf(“%d”,&a);

→int型の変数aに値を入力する。

変数の前に&を付けるのを忘れないように！

- 変換指定文字一覧

double型は%fでなく%lfなので注意

変換指定文字	使う型
%d	int型
%lf	double型
%s	char型(文字列として出力)
%c	char型(1文字だけ出力)

練習問題１

- int型の値を読み取って出力するプログラムを作れ
- 答え

```
#include <stdio.h>
int main(void){
    int a;
    scanf(“%d”,&a);
    printf(“a=%d¥n”,a);
    return 0;
}
```

if文 –条件分岐- (1)

- 条件分岐

```
if(a>=10){
 printf(“aは10以上です。”);
}
```

→aの値が10以上の時に出力される。

条件式が正しい時に中の文が実行される！

if文 –条件分岐- (2)

- 演算子について

演算子	意味
a==b	aとbが等しい
a!=b	aとbが等しくない
a>b	aがbより大きい
a>=b	aがbより大きいか等しい
a<b	aがbより小さい
a<=b	aがbより小さいか等しい

if文 –条件分岐- (3)

- else文

条件が成り立たないときに実行する。

```
if(a>=10)
    printf(“aは10以上です。¥n”);
else
    printf(“aは10未満です。¥n”);
```

こんな感じ

if文 –条件分岐- (4)

- else if文

条件が成り立たないときの条件分岐

```
if(a>=20)
    printf(“aは20以上です。¥n”);
else    if(a>=10)
    printf(“aは10以上20未満です。¥n”);
else
    printf(“aは10未満です。¥n”);
```

こんな感じ

練習問題2

- 2つの値をそれぞれa,bに読み取ってaとbの関係を
aがbより大きいか、等しいか、小さいか判断して出力せよ。
- 答え

```
scanf("%d%d",&a,&b);
if(a>b)
    printf("aはbより大きいです。¥n");
else if(a==b)
    printf("aはbと同じです。¥n");
else
    printf("aはbより小さいです。¥n");
```

while文 –繰り返し-

- 繰り返し

```
int a=0;
while(a<10){
    printf(“a=%d¥n”,a);
    a++;
}
```

→a=0

a=1

…

a=9

for文 –繰り返し-

- 繰り返し

```
for(a=0;a<10;a++)
    printf("a=%d¥n",a);
}
```

→結果は同じ

for文は(初期化;繰り返し条件;処理が終わるたびに実行ｽﾙ式)で成り立っている。

ただし、これらは省略可能

明日習うC++では初期化式で変数を宣言できたりしちゃう

例) for(int i=0;i<10;i++)→iはfor文の中のローカル変数になる。

switch文 –多方向分岐-

- 分岐

```
switch(a){
    case 1 : printf(“aは1です。¥n”);break;
    case 5 : printf(“aは5です。¥n”);break;
    default : printf(“aは1でも5でもないです。¥n”);break;
}
```

定数式の後はコロン(:)を付ける！

breakを忘れると次の処理に移行してしまうので注意！

- どんな時に使う？

else ifでいっぱい条件分岐させるのが面倒なときに使う

breakとcontinue

```
for(a=0;a<10;a++){
    if(a==1)continue;
    if(a==3)break;
    printf("aは%dです。¥n",a);
}
```

aが0,2の時は出力されるが、他は出力されない。

1の時は出力されずにスキップされる。

3の時は繰り返しを抜けだしてしまい

残りの処理を行わない

練習問題3

- 値を10回読み込み、そのまま出力されるプログラムを作れ。ただし、マイナスの値が入力された時はスキップし、0が入力された時は終了させること

- 答え

```
for(i=0;i<10;i++){
    scanf(“%d”,&a);
    if(a<0)continue;
    else if(a==0)break;
    printf(“%d¥n”,a);
}
```

配列(1)

- 複数の変数を一気に宣言できる。

int a[5];

→a[0]～a[4]まで使えるように

- 配列の初期化

a[5]={0}; //a[0]~a[4]に0を代入する。

a[5]={1}; //a[0]=1 a[1]~a[4]に0が代入される。

a[]={1,2,3,4,5}; //a[0]=1 a[1]=2 … a[4]=5 と代入される。

※以下の場合はエラーになるので注意

a[2]={1,2,3,4}; //要素数より代入する要素が多い

int a[5];　a[]={1,2,3,4,5} ;

//このような形の初期化は宣言時のみ

配列(2)

- 2次元配列

a[3][3]

→

a[0][0]	a[0][1]	a[0][2]
a[1][0]	a[1][1]	a[1][2]
a[2][0]	a[2][1]	a[2][2]

- 配列はfor文と、2次元配列は
  2重for文と組み合わせるといい

- なお、3次元配列以上も出来る。

a[3][3][3]/a[3][3][3][3] etc…

構造体

- struct Student{
  Int　number;
  int　point;
  };　←ここセミコロン付ける
- 複数のデータをまとめる時に便利
  struct Student a,b;
  a.number=1; a.point=50;の形でアクセス

- C++では構造体の発展形であるクラスが登場
  非常に重要な概念になってくる。

練習問題4

- 生徒のNoと数学、英語の点数と平均を格納する構造体を作り、生徒3人の数学、英語の点数を入力して、各生徒のNo、数学、英語の点数と平均点を出力せよ。

- 答え

```
struct Student{
    int no;
    int math;
    int english;
    int average;
};
```

練習問題4

- 答え(続き)

```
int main(void){
    struct Student a[3];
    int i;
    for(i=0;i<3;i++){
        a[i].no=i;
        scanf("%d%d",&a[i].math,&a[i].english);
        a[i]. average =(a[i].math+a[i].english)/2;
    }
    for(i=0;i<3;i++){
        printf("no:%d 数学:%d 英語:%d 平均%d ¥n",
        a[i].no, a[i].math, a[i].english, a[i]. average);
    }
    return 0;
}
```

関数(1)

- 処理を複数のプロセスに分ける。
- 同じような処理を複数回繰り返す時に便利
- 例)2乗した後に2倍する関数

```
int cal(int a){
    a=(a*a)*2;
    return a;
}
Int main(void){
    int a=10;
    int b=cal(a);
    printf("%d",a);
    return 0;
}
```

関数(2)

- 関数の形式

戻り値の型 関数名(引数){

　　実際の処理

　　return 戻す値

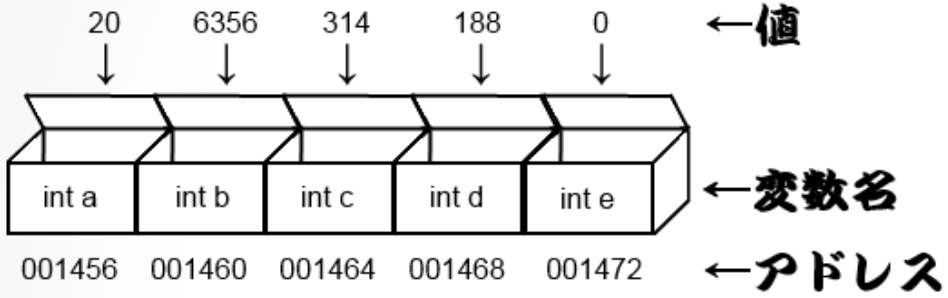
}

- main文にもこれは当てはまる。
- 何も返さない場合にはvoid型を使う。

return文は書かなくても構わない

ポインタ(1)

- 簡単に言うと変数がしまわれている場所を参照できる。



•宣言の方法は 変数の形名 *変数名

例)int *p;

•普通の変数に&を付けるとアドレスを参照できる。

Int a;

p=&a;

このように代入できる。

ポインタ(2)

- ポインタの使い方の例

```
Int main(void){
    int a=20;
    int *p=&a;
    *p+=20;
    printf(“*p:%d¥n”,*p);
    printf(“a:%d¥n”,a);
}
```

→両方とも40が出力される。

ポインタ(3)

- ポインタのまとめ

変数	値	アドレス
int a;(int型の変数)	a	&a
int *p;(int型のポインタ)	*p	p

- ポインタのポインタ

ポインタ自身にもアドレスはある。ポインタ変数で
printf("%d",&p);でポインタのアドレスを参照できる。
ポインタのポインタを作るにはint **q;のようにする。
ちなみにポインタのポインタのポインタもできる。

ポインタ(4)

- 配列はポインタである。
- char a[5];を宣言した時、aというポインタから始まる変数を連続で５つ宣言したことと等しいことになる。
- すると&a[0]とaは等しくなるため
  &a[1]=a+1 , &a[2]=a+2のようになる。

ポインタ		アドレス(例)
a	&a[0]	1000
a+1	&a[1]	1001
a+2	&a[2]	1002
a+3	&a[3]	1003

練習問題5

- 2つのポインタ変数を受け取り、それぞれの値を2倍にした上入れ替える関数void dswap(int *a,int *b)を作れ。
- 答え

```
void dswap(int *a,int *b){
    int sw=*a;
    *a=*b;
    *b=sw;
    *a*=2;
    *b*=2;
}
```

列挙型

- 列挙型

定数のリストを定義することが出来る。

```
enum Week{Sunday, Monday, Tuesday};
int main(void){
    enum Week w1=Sunday;
    if(w1==Sunday){
        printf("%d¥n",w1);
    }
    returun 0;
}
```

→0

実はint型の値にそれぞれ定数をつけているだけ
値は左から0,1,2とされる。

おしまい

明日はいよいよC++の勉強をやります。
場所は変わって0306になるので注意！