C言語講座第４回

○　ファイル入出力

ファイルの入出力流れ：
①FILE 型変数（ポインタ）を作る
②fopen 関数でファイルを開く
（第二引数が"r"だと読み込み。"w"だと書き込み。"r"はファイルが存在していないといけないが、"w"は無かったら勝手に作る。）

③fprintf や fscanf 関数で書き込んだり読み込んだりする
④fclose 関数でファイルを閉じる（必須）

※
ファイルはすべて FILE 型のポインタ変数を介して扱う。

```
#include <stdio.h>

int main(void)
{
        FILE *fp;
        char writestr[200],readstr[200];

        //ファイルへの書き込み
        printf("ファイルにデータを書き込みます。文字列を入力してください¥n");
        scanf("%s",writestr);
        fp = fopen( "test.txt" , "w");    //書き込みモードでファイルを開く
        fprintf( fp , "%s" , writestr );  //ファイルに対するprintf
        fclose(fp);                       //ファイルを閉じる

        //ファイルからの読み込み
        fp = fopen( "test.txt", "r");     //読み込みモードでファイルを読み込む
        if( fp == NULL )
        {
                return -1;        //読み込みに失敗したら終了
        }
        fscanf( fp , "%s" , readstr );
        printf("test.txtに書かれているデータは¥n%s¥nです。¥n",readstr);
        fclose(fp);
        return 0;
}
```

○構造体とクラス

※サンプルコードは次ページ
構造体・クラスは複数の変数をまとめて扱う方法。
Ｃの機能にあるのは構造体。クラスはＣ言語ではなくＣ＋＋の記述法ではあるが、
基本的な使い方がかなり近い上、今後役立つと思われるためここに書いておく。
構造体は struct、クラスは class と記述してから扱いたい変数を記述していく。（下記）

```
struct student {
                char name[30];
                int number;
};
```

```
class CStudent
{
public: //←クラスの場合はこれが必要
                char name[30];
                int number;
};
```

この場合、名前(name)と番号(number)を持てる自作の型を作ったようなもの。
struct student 変数名（CStudent 変数名）のように変数を作り、「.」を用いて使いたい要素にアクセスする。
例えば、number を使いたければ、「変数名．number 」と書く。

なお、クラスを使う場合はファイルの拡張子を．ｃではなく．ｃｐｐとしないといけない。

※下記２つのコードは同じことをしている

```
//構造体（C）
#include <stdio.h>

struct student {
                char name[30];
                int number;
};

int main(void)
{
        struct student data[5];  //構造体の変数の宣言
        int i;
        for( i = 0 ; i < 5 ; i++ )
        {
                data[i].number = i + 1;
                printf("名前を入力¥n->");
                scanf("%s", &data[i].name);
        }
        for( i = 0; i < 5 ; i++ )
        {
                printf("No.%d¥n名前：%s¥n",data[i].number,data[i].name);
        }
        return 0;
}
```

```
//クラス（C++）
#include <stdio.h>

class CStudent
{
public:
                char name[30];
                int number;
};

int main(void)
{
        CStudent data[5];       //クラス
        int i;
        for( i = 0 ; i < 5 ; i++ )
        {
                data[i].number = i + 1;
                printf("名前を入力¥n->");
                scanf("%s", &data[i].name);
        }
        for( i = 0; i < 5 ; i++ )
        {
                printf("No.%d¥n名前：%s¥n",data[i].number,data[i].name);
        }
        return 0;
}
```

○　ゲーム製作へ向けて

①　乱数

コンピューターの発生させる乱数は完全にランダムな乱数でではなく、なんらかの計算によって乱数を生成している。(擬似乱数という)。
そのため stdlib.h にある rand 関数を呼んでもそれだけでは毎回同じ乱数が生成されてしまう。
乱数にはそれを発生させる計算元の数（乱数の種）があるため。そこを変えてやれば、違う値が生成される。
さらにこの乱数の種に現在時刻等の刻々と変化する量を使うことで毎回違う数が生成されることになる。

```
#include <stdio.h>
#include <time.h>  //現在時刻の取得に必要
#include <stdlib.h>  //rand(),srand()関数の使用に必要

int main(void)
{
        int i;
        srand( (unsigned int)time(NULL) );//現在時刻を乱数の種に
        for(i = 0 ; i < 10 ; i++)
        {
                printf("%d¥n",rand()%100 );//100未満の乱数を表示
        }
        return 0;
}
```

②　リアルタイム入力

・　kbhit()
キーボードが押されていたら１、いなかったら０。
if 文に渡すと押されていたら処理を実行する。

・　getch()
押されたキーを得る。

※
Enter や Escape は右のサンプルコードで押すと変な風に表示されるが、%c を%d に変えることでこれらのキーにも数字が割り当てられていることがわかる。
この数字を比較することで Enter や Escape を使うことも可能。

```
#include <stdio.h>
#include <conio.h>        //kbhit(),getch()

int main(void)
{
        int i;
        char input;       //入力受付
        while(1) //無限ループ
        {
                if( kbhit() )     //キーボードが押されたら
                {
                        input = getch(); //押されたキーを取得
                        if( input == 'e' )//eが押されていたら終了
                        {
                                printf("終了します¥n");
                                break;
                        }
                        printf("押されたキーは「%c」です¥n",input);
                }
        }
        return 0;
}
```