情報処理演習
（11）テキストファイルの入出力

知能システム学　准教授
万 偉偉（ワン ウェイウェイ）

これからの予定

- 12月20日　ポインタ・ファイル　14時30分まで
  - 課題20日と一緒提出期限1月17日・24日まで伸びる
- 12月27日　ファイル・あみだくじ
  - 課題（20日と一緒）
  - 一回提出1月17日
  - 2回提出1月24日
  - あみだくじを提出する必要はありません
- 1月24日　最終課題
  - 1月31日提出
  - 一回のみ
  - 救済措置なし

テキストファイル入出力

- メモリのアドレスを値として定義された変数
  - 型に強く縛られる
  - 参照、増減
- 配列の名前はポインタ
  - 配列を引数とした関数
  - 関数から修正したデータを貰う手法
    - 返り値
    - 参照渡し
- 多重ポインタ・多次元配列

テキストファイル入出力

- 数値，文字列といったファイル中のデータを読み込んで（入力して）加工する，加工したデータをファイルに書き出し（出力し）たい場合は多くある．
- ストリーム
  - C言語では数値や文字列の文字列の入出力にはストリーム（文字が流れる川のようなもの、メモリの一部）を使用する．**ストリームはFILE型変数（厳密には構造体）によって制御され**，ファイルの読みこみ・書き出しはFILE型へのポインタ型を宣言し，そのアドレスを読み込み・書き出しを行いたいファイル名のものに指定することにより実現される．

ストリーム

- 例えば我々がこれまで使用したscanf（入力）やprintf（出力）といった関数では，プログラムの開始時に準備される標準入出力ストリーム（stdinやstdout）を通し文字や数値の入出力を行っており，それらの**ストリームのFILE型変数**のアドレスがそれぞれキーボードとディスプレイに割り当てられていることにより，キーボードからの入力とディスプレイ出力を実現している．

FILE *fpの中身

- FILE *fpは，構造体FILEへのポインタ

stdio.h

- NULL：値のないポインタを意味する．void*型で値は0
- EOF (End of file)：ファイル末尾を表す (-1)

ファイル入力例

```
#include <stdio.h>
int main(){
	//ファイル変数(構造体)へのポインタを宣言
	FILE *fp_in;
	double sum=0.0;
	double temp;
	//ファイルを読み込みモードで開く
	fp_in = fopen("sample_data.txt","r");
	//ファイルオープンに失敗した場合
	if(fp_in==NULL){
		//失敗と表示し終了
		printf("ファイルオープンに失敗しました\n");
		//エラー終了の場合戻り値を0以外にする
		return 1;
	}
	//fscanfが正常終了する間読み込みを繰り返す．
	while( fscanf(fp_in, "%lf" , &temp) == 1 ) {
		sum+=temp;
	}
	//ファイルを閉じる
	fclose(fp_in);
	printf("ファイル内の数値の総和は%fです\n",sum);
	return 0;
```

ポイント１

- ファイルの入出力のためにはまずFILE型のポインタを宣言する．
  - FILE *fp_in;
- そののちに fp_in=fopen(″ファイル名″，″読み書きについてのオプション″) として，fopen関数にて入出力に使用するファイル名，データを読むのか，書くのかのオプションを指定してファイルを開く．
- 例えば上のプログラム内
  - fp_in = fopen(″sample_data.txt″,″r″);
  - sample_data.txtという名前のファイルを ″r″（読みとり）モードで開くをことを意味する．

ポイント２

- ファイルを開くのに失敗した場合 fopen() は NULL文字を返す，これを利用して上のプログラムではファイルが読み込めなかった場合は異常終了としてreturn 1 返すようにしている．

```
if(fp_in==NULL){
    //失敗と表示し終了
    printf("ファイルオープンに失敗しました¥n");
    //エラー終了の場合戻り値を0以外にする
    return 1;
}
```

ポイント３

- ファイルを "r" オプション指定で開いた状態であれば，fscanf関数を用いてファイルから一行読みとって数値を変数に保存することができる．
- fscanfの使い方はscanfとほぼ同じであり，違いは fscanf(fp_in, "%lf" , &temp) からわかるように読み込みたいファイルのFILE型変数のアドレス（この場合 fp_in）を指定することだけである．

ポイント４

- fscanf関数は２回，３回，４回，...，と繰り返し実行すると自動的にファイル内２行目，３行目，４行目，...，を読み込む仕様になっており，また戻り値はその行で変数の値を読み込んだ回数となる．
- これを利用してプログラム内の while(fscanf(fp_in, "%lf" , &temp) == 1) では，fscanf(fp_in, "%lf" , &temp) の戻り値が１になる限り(読み込む値がある限り)
- 条件が真となるのでテキストファイルの最後までfscanfを繰り返す仕組みになっている．

ポイント５

- 一度開いたファイルは処理が終わったら必ずfclose(fp_in);にて閉じる必要がある．忘れがちなので注意すること．

ファイル出力例

```
#include <stdio.h>
#include <math.h>
int main(void){
    //ファイル変数(構造体)へのポインタを宣言
    FILE *fp_out;
    char str[256];
    double input;
    printf("ファイル名を入力してください\n");
    scanf("%s",str);
    //ファイルを書き込みモードで開く
    fp_out = fopen(str,"w");
    //ファイルオープンに失敗した場合
    if(fp_out==NULL){
        //失敗と表示し終了
        printf("ファイルオープンに失敗しました\n");
        //エラー終了の場合戻り値を0以外にする
        return 1;
    }
    while(1){
        printf("正の値を入力してください\n");
        printf("終了する場合は負の値または0を入力してください\n");
        scanf("%lf", &input);
        if(input <=0 ) break;
        //ファイルに受け取った値とそのlogの値を書き出す．
        fprintf(fp_out, "%f %f\n",input,log(input));
    }
    fclose(fp_out);
    return 0;
}
```

ポイント１

- ファイルへのデータ書きだしは，さきほどfopen関数のオプションを “w” に変えるだけでOKである．“w”とは新規ファイルを作成する（ファイル名が存在すれば上書する） ことである．
  - 例えばプログラム内では scanf("%s",str) で文字配列としてファイル名を読み込み，FILE型ポインタ fp_outに対して，fp_out = fopen(str,"w");とすることにより書き出すファイルを開いている．
- 書き出しのオプションについては書き出す "w" 以外にも，既存のファイルの最後に追加して書き出す "a" を使用することもできる．

ポイント２

- 書き出しオプションにてファイルを開いた状態であればfprintf関数を用いて値を書きだすことが可能である．
- こちらも使い方はprintfとほぼ同じであるが，第一項に開いたファイルのFILE型のアドレスを追加する必要がある．
- fscanf同様にfprintf関数は２回，３回，４回，...，と繰り返し実行すると自動的にファイル内２行目，３行目，４行目，...，に書き込む仕様になっている．