電気情報系実験3

プログラミング例

～fopen fclose Makefile etc...～

2017年10月18日

M17J4059Y 安場　裕人

Makefile

```
CC = gcc
CFLAGS = -g -Wall
PROGRAMS = input_data.o count_data.o output_data.o main.o main

all: $(PROGRAMS)

clean:
	rm -f *.o *~

input_data.o: input_data.c parameter.h
	$(CC) $(CFLAGS) -c $^

count_data.o: count_data.c parameter.h
	$(CC) $(CFLAGS) -c $^

output_data.o: output_data.c parameter.h
	$(CC) $(CFLAGS) -c $^

main.o: main.c parameter.h
	$(CC) $(CFLAGS) -c $^

main: main.o input_data.o count_data.o output_data.o
	$(CC) $(CFLAGS) -o $@ $^
```

parameter.h

```
/*****************************
name; network.h
abstruct; network modules header
date; 2017/10/16
writer; Yaseba Hiroto

using packages;
  standard.h

fixed date;

 *****************************/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

/***********************************************
        define numbers
***********************************************/
#define MAX_NAME 30
#define MAX_DATA 20
#define MAX_STR 10

/***********************************************
        define structure
***********************************************/
typedef struct Data{
  char first[MAX_STR];
  char second[MAX_STR];
  int third;
} DATA;
```

main.c

```
/*****************************
name; main.c
abstruct; test program (function fopen fclose)
                    (and cross compile)
date; 2017/10/16
writer; Yaseba Hiroto

using packages;
  test.h

fixed date;

 *****************************/
#include "parameter.h"

int main(int argc, char *argv[]){12

  //incorrect calling alert
  if(argc != 2){
    printf("main filename.txt.\n");
    return -1;
  }

  //variable define
  //for filename
  char file[MAX_NAME];
  //structure of data
  DATA data[MAX_DATA];
  //counter of data
  int count;

  //function define
  void input_data(DATA * data, char * filename); //from input_data.c
  int count_data(DATA * data); //from count_data.c
  void output_data(DATA * data, int c); //from output_data.c

  //catch a filename
  if((strcpy(file, argv[1])) == NULL) return -1;

  //catch filename's data
```

```
  input_data(data, file);

  //count data
  count = count_data(data);
  //output filename' data
  output_data(data, count);

  //end of program
  return 0;
}
```

input_data.c

```
#include "parameter.h"

/*
name; input_data
abstract; this input filename's data to DATA[]
expected data;
char * first \t char * second \t int third
char * first \t char * second \t int third
...
...
...

date; 2017/10/16
writer; Yaseba Hiroto

input; DATA * data
       char * filename
output; DATA * data

fixed date;
*/
void input_data(DATA * data, char * filename){
  //define variable
  //for roop
  int i;

  //file pointer
  FILE *fp;
  //temps
  char temp_first[MAX_STR];
  char temp_second[MAX_STR];
  int temp_third;

  //initialization temp_first
  //strcpy(temp_first, "NULL");

  //open file
  if((fp = fopen(filename, "r")) == NULL) {
    printf("you can't open file\n");
    return;
```

```
    }

  //roop to file end or MAXDATA
  //catch data to temps by fscanf
  for(i = 0; (((fscanf(fp,"%s\t%s\t%d\n", temp_first, temp_second, &temp_third)) != -1) &&
                                                                    (i < MAX_DATA)) ; i++){

  //copy data to DATA * data
     strcpy(data[i].first, temp_first);
     strcpy(data[i].second, temp_second);
     data[i].third = temp_third;
  //end roop
  }

  //insert last data
  //first NULL
  //scond NULL
  //third 0
  strcpy(data[i].first,"NULL");
  strcpy(data[i].second,"NULL");
  data[i].third = 0;

  //file close
  fclose(fp);

  //function catch end
}
```

count_data.c

```
#include "parameter.h"

/*
name; count_data
abstract; this count data in DATA structure
date; 2017/10/16
writer; Yaseba Hiroto
packaged in; test.c

input; DATA * data
output; int counter;

fixed date;
*/
int count_data(DATA * data){
  //define variable
  //for roop
  int i;

  //counter
  int counter = 0;

  //roop to find first is NULL or MAX_DATA
  for(i = 0; ((strcmp(data[i].first, "NULL") != 0) && (i < MAX_DATA)); i++){

  //counter increment
    counter++;

  //roop end
  }

  //program end return counter
  return counter;
}
```

output_data.c

```
#include "parameter.h"

/*
name; output_data
abstract; this output data, repeated to c
output form
page|first     |second    |third|
---------------------------------
   1|first     |second    |third|
---------------------------------
   2|first     |second    |third|
---------------------------------
...
...

date; 2017/10/16
writer; Yaseba Hiroto

input; DATA * data
       int c
output; void

fixed date;
*/
void output_data(DATA * data, int c){
  //variable define
  //for roop
  int i, n;
  //width;
  int width;
  width = 4 + 1 + 10 + 1 + 10 + 5 + 1;

  //ceiling
  printf("page|first     |second    |third|\n");
  for(i = 0; i < width; i++){
    printf("-");
  }
  printf("\n");

  //data writing roop to c or MAX_DATA
  for(i = 0; (i < c) && (i < MAX_DATA); i++){
```

```
    printf("%4d|%10s|%10s|%5d\n",i+1 ,data[i].first, data[i].second, data[i].third);
    for(n = 0; n < width; n++){
      printf("-");
    }
      printf("\n");
  //data writing roop end
  }

  //inform data end
  printf("data end\n");

  //function end
}
```

move.sh

```
#!/bin/sh

#test using
rm result.txt
make clean
make main

./main data.txt > result.txt

cat result.txt
```

data.txt

```
jack     kkk     12
jon      ggg     37
bbb      west    49
koji     quort   89
clover   mike    90
```

result.txt

```
page|first     |second    |third|
---------------------------------
   1|      jack|       kkk|   12
---------------------------------
   2|       jon|       ggg|   37
---------------------------------
   3|       bbb|      west|   49
---------------------------------
   4|      koji|     quort|   89
---------------------------------
   5|    clover|      mike|   90
---------------------------------
data end
```

外部仕様書
テストプログラム（Test_V2）

安場裕人

2017/8/30

1　概要

この外部仕様書は電気情報系実験受講者向けに作成されたプログラムの外部設計を示す。このプログラムの目的は以下の項目を受講者が理解するために作成された。

- fopen, fclose
- Makefile
- 分割コンパイル
- .sh ファイル
- コマンドライン引数

2 節では、パッケージ test_V2 内のファイル、ディレクトリの役割を示す。3 節では main プログラムとそれに関わるサブルーチンについて示す。

なお、プログラムの内部設計については別紙に示す。

2　パッケージ：test_V2

パッケージ test_V2 は 1 節で挙げた項目を用いたプログラムを実現するために作成された。ソースコードファイル、ヘッダーファイル、データファイル、仕様書類の入ったディレクトリが一括でまとめられている。

2.1　Makefile

分割して作成されたプログラムを結合してコンパイルする。コンパイルには gcc コンパイラを使用する。デバッグ用フラグとして-g-Wall を使用する。

2.2　data.txt

main プログラムで読み込まれるデータファイル。形式はテキスト形式。

想定している data.txt の書式を以下に示す。読み込みを想定しているファイルの書式を以下に示す。

string11 \t string12 \t integer1 \n

string21 \t string22 \t integer2 \n

...

ここで、string**は文字列を示す。また integer*は整数を示す。例を以下に示す。

jack mathmatic 97

pole scientic 83

...

2.3 parameter.h

main プログラムで使用する構造体 DATA の宣言を行う。またメモリ使用量を決定する変数の宣言を行う。main プログラムで使われるパッケージの include を行う。

2.4 result.txt

main プログラムを書き込むためのファイル。形式はテキスト形式。

2.5 Move.sh

前回実行情報の削除から main プログラムのコンパイル、mian プログラムの実行、結果ファイルの表示までを一括で行うシェルファイル。

2.6 dist

仕様書類の図をまとめたディレクトリ。

2.7 dist_tex

仕様書類をまとめたディレクトリ。仕様書類は tex を用いて作成されている。tex ファイルのコンパイル、pdf に変換するためのシェルファイルも同封されている。

2.8 main.c

test_V2 のメインプログラムのソースコード。概要は 3 節に示す。コンパイルには以下のファイルを要する。

- parameter.h
- main.c
- input_data.c
- count_data.c
- output_data.c

2.9 input_data.c

サブルーチン 1 のソースファイル。概要は 3.1 節に示す。

2.10 count_data.c

サブルーチン 2 のソースファイル。概要は 3.2 節に示す。

2.11 output_data.c

サブルーチン 3 のソースファイル。概要は 3.3 節に示す。

3 main プログラム

main プログラムはファイルの読み出し、ページ数のついた表の形式で出力する。コマンドライン引数として、読み出すデータファイルを指定する。図 1 に各サブルーチンとメインプログラムの役割を示す。

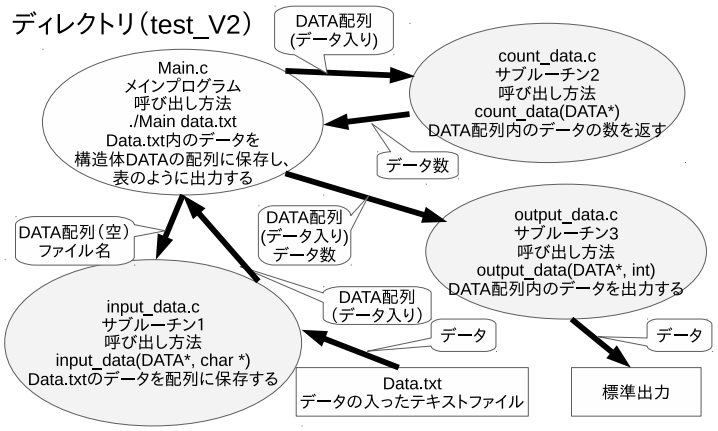

図 1: プログラムの相関図

以下に main プログラムの手順を示す。

- 1 ：サブルーチン input_data を用いて data.txt のデータを読み出す。
- 2 :サブルーチン count_data.c を用いて読みだしたデータの数を数える。
- 3 ：ブルーチン output_data を用いて読みだしたデータをその数を出力する。

入力はテキスト形式のファイルから行う。出力は標準出力を用いて行う。

3.1　サブルーチン１

入力されたファイル名のファイルからデータを読み出し、DATA 型の配列に格納する。

- 引数１：DATA * (データを格納する配列のポインタ)
- 引数２：char * (データを読み出すファイルの名前を格納した配列のポインタ)
- 返値　：なし

3.2　サブルーチン２

データ配列内に格納されているデータの数を数える

- 引数：DATA * (データが格納されている配列のポインタ)
- 返値：int (データの数)

3.3　サブルーチン３

- 引数１：DATA * (データが格納されている配列のポインタ)
- 引数２：int (データの数)
- 返値　：なし

内部仕様書
テストプログラム（test_V2/main.c）

安場裕人

平成 29 年 12 月 6 日

1　概要

この仕様書は test_V2/main.c の内部設計を記載する。main.c は test_V2 のプログラムの main ルーチンである。機能としては、data.txt の内容を読み出し、そのデータ数を数える。そして、表形式で標準出力する。一般的な情報を以下に示す。

- 名称：main.c
- 作成者：安場　裕人
- コマンドライン引数： データファイルの名前
- 返値：なし

main.c はヘッダーファイルとして parameter.h を使う。また、以下のサブルーチンを使用する。

- input_data.c
- count_data.c
- output_data.c

図 1 に main.c のフローチャートを示す。

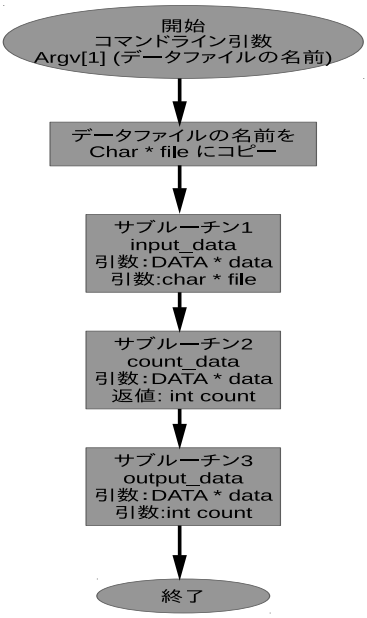

図 1: main.c のフローチャート

内部仕様書
テストプログラム（test_V2/input_data.c）

安場裕人

平成 29 年 12 月 6 日

1　概要

この仕様書は test_V2/input_data.c の内部設計を記載する。input_data.c は test_V2 の main ルーチンのサブルーチン 1 に相当する。機能としては、与えられたファイル名のファイルからデータを読み出し、構造体 DATA 配列に格納する。一般的な情報を以下に示す。

- 名称：input_data.c
- 作成者：安場　裕人
- 引数 1 ：DATA * data (読み込んだデータを格納する配列のポインタ)
- 引数 2 ：char * filename (読み込むファイルの名前を格納した配列のポインタ)
- 返値：なし

読み込みを想定しているファイルの書式を以下に示す。

```
string11\tstring12\tinteger1\n
string21\tstring22\tinteger2\n
...
```

ここで、string**は文字列を示す。また integer*は整数を示す。例を以下に示す。

```
jack   mathmatic    97
pole   scientic     83
...
```

構造体 DATA は parameter.h 内で宣言されている。また、一つの文字列の文字数の上限として、parameter.h 内で宣言されている MAX_STR の値を使う。データの行数の上限として、parameter..h 内で宣言されている MAX_DATA の値を使う。それを超えた場合は各上限数のみを構造体 DATA に格納する。

図 1 に input_data.c のフローチャートを示す。

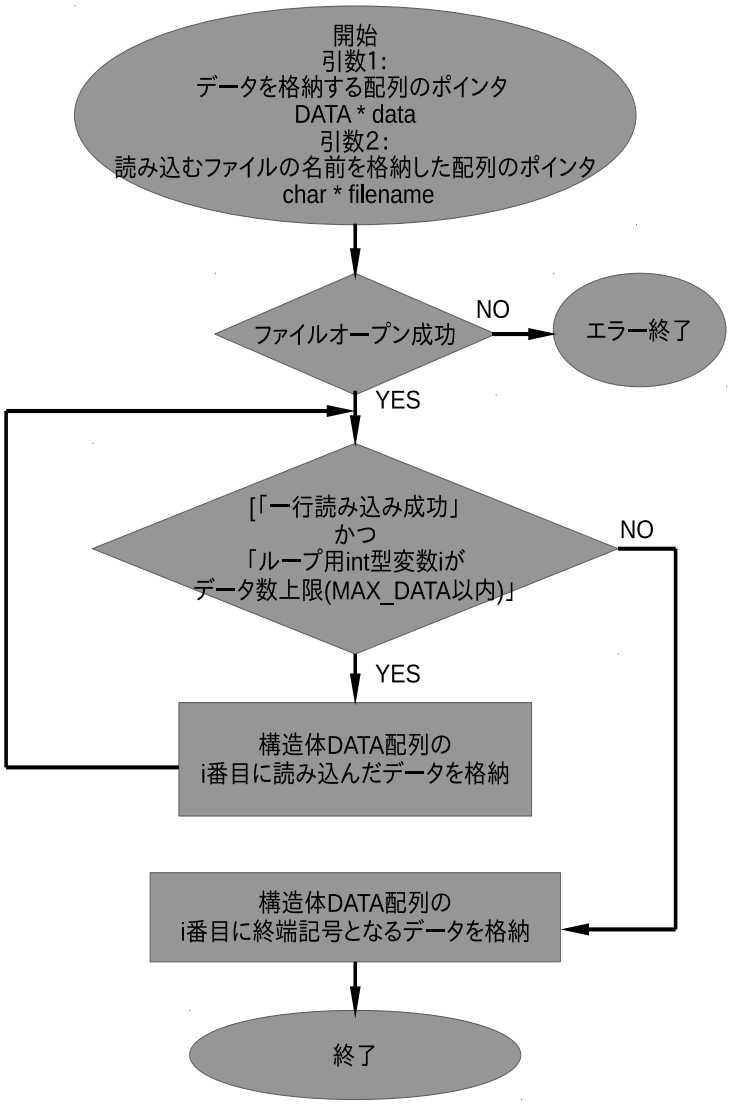

図 1: input_data.c のフローチャート

内部仕様書
テストプログラム（test_V2/count_data.c）

安場裕人

平成 29 年 12 月 6 日

1　概要

この仕様書は test_V2/count_data.c の内部設計を記載する。count_data.c は test_V2 の main ルーチンのサブルーチン 2 に相当する。機能としては、構造体 DATA 配列の使われている数を数える。一般的な情報を以下に示す。

- 名称：count_data.c
- 作成者：安場　裕人
- 引数：DATA * data (数を数えるデータ配列)
- 返値：int counter (DATA 配列内のデータの数)

構造体 DATA は parameter.h 内で宣言されている。また、counter は parameter.h 内で宣言されている MAX_DATA の値を上限とし、それを超えた場合は MAX_DATA の値が返値としてかえされる。

図 1 に count_data.c のフローチャートを示す。

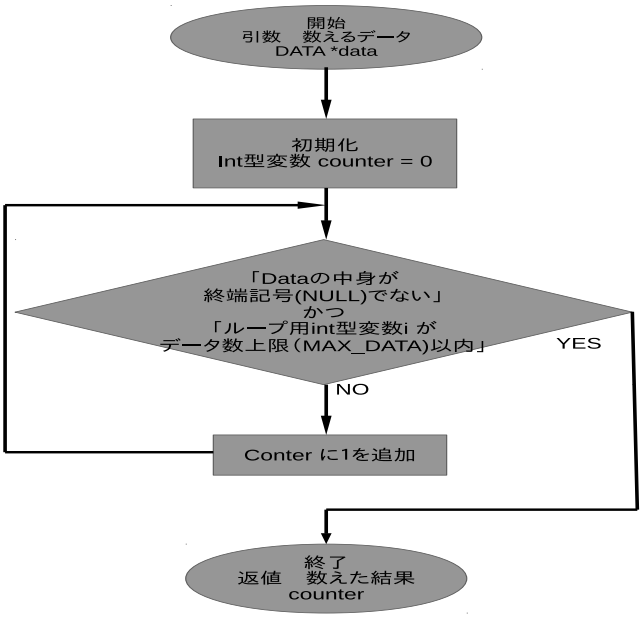

図 1: count_data.c のフローチャート

内部仕様書
テストプログラム（test_V2/output_data.c）

安場裕人

平成 29 年 12 月 6 日

1　概要

この仕様書は test_V2/output_data.c の内部設計を記載する。output_data.c は test_V2 の main ルーチンのサブルーチン 3 に相当する。機能としては、構造体 DATA 配列内のデータを表の形で出力する。一般的な情報を以下に示す。

- 名称：output_data.c
- 作成者：安場　裕人
- 引数 1 ：DATA * data (出力するデータ)
- 引数 2 ：int count (構造体 DATA 配列内に存在するデータの数)
- 返値：なし

出力形式を以下に示す。

```
page|first     |second    |third|
---------------------------------
   1|  string11|  string12| int1|
---------------------------------
   2|  string21|  string22| int2|
---------------------------------
  ...
---------------------------------
data end
```

ここで、string**は文字列を示す。また int*は整数を示す。例を以下に示す。

```
page|first     |second    |third|
---------------------------------
   1|      jack| mathmatic|   97|
---------------------------------
   2|      pole|  scientic|   83|
---------------------------------
  ...
---------------------------------
data end
```

構造体 DATA は parameter.h 内で宣言されている。また、表示する数は parameter.h 内で宣言されている MAX_DATA の値を上限とし、それを超えた場合は MAX_DATA の数だけを表示する。

図 1 に output_data.c のフローチャートを示す。

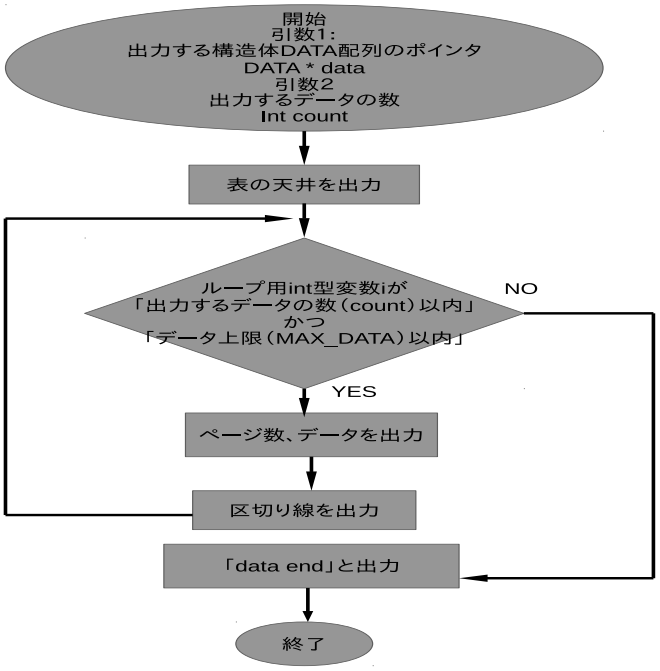

図 1: output_data.c のフローチャート

内部仕様書
テストプログラム（test_V2/その他）

安場裕人

平成 29 年 12 月 6 日

1　parameter.h

この仕様書は test_V2/parameter.h の内部設計を記載する。test_V2 ディレクトリ内のソースファイルの全てで用いられるヘッダーファイルである。parameter.h 内では、ライブラリの include、文字列や配列を宣言する際の領域を確保する数の宣言、構造体の宣言を行なっている。一般的な情報を以下に示す。

- 名称：parameter.h
- 作成者：安場　裕人
- ライブラリ１：stdio.h
- ライブラリ２：string.h
- ライブラリ３：stdlib.h

宣言される変数の使用目的を以下に示す。

- MAX_NAME：ファイル名の文字数の上限
- MAX_DATA：データ数の上限
- MAX_STR：データ内の文字列の文字数の上限

1.1　構造体 DATA について

構造体 DATA は data.txt から読みだしたデータを文字列と整数に分けて保存するための構造体である。構造体内の変数について、以下に説明する。

- char first[MAX_STR]：文字列一個目
- char second[MAX_STR]：文字列二個目
- int third：整数

2 Makefile

この仕様書は test_V2/Makefile の内部設計を記載する。test_V2 ディレクトリ内のソースファイルのコンパイルを行うための Makefile である。一般的な情報を以下に示す。

- 名称：Makefile
- 作成者：安場　裕人
- コンパイラ: gcc
- フラグ: -g -Wall

なお、フラグはデバグの際に用いる。
以下に項目の説明を行う。

- all：全てのオブジェクトファイル、実行ファイルを作成
- clean：全てのオブジェクトファイル、実行ファイルを削除
- *.o：*.c のソースファイルをヘッダファイル parameter.h とともにコンパイルし、王ジェクトファイル*.o を作成。
- main メインのソースコードをコンパイルし、実行ファイル main を作成。

3 data.txt

この仕様書は test_V2/data.txt の内部設計を記載する。data.txt は test_V2 のメインプログラムで読み込むデータファイルである。一般的な情報を以下に示す。

- 名称：Makefile
- 作成者：安場　裕人
- 形式：テキスト形式

data.txt の書式を以下に示す。

```
string11\tstring12\tinteger1\n
string21\tstring22\tinteger2\n
...
```

ここで、string**は文字列を示す。また integer*は整数を示す。例を以下に示す。

```
jack   mathmatic    97
pole   scientic     83
...
```

4 result.txt

この仕様書は test_V2/result.txt の内部設計を記載する。result.txt は test_V2 のメインプログラムの出力を保存するためのファイルである。一般的な情報を以下に示す。

- 名称：Makefile
- 作成者：安場　裕人
- 形式 : テキスト形式

result.txt 内に保存される書式を以下に示す。

```
page|first     |second    |third|
---------------------------------
   1|  string11|  string12| int1|
---------------------------------
   2|  string21|  string22| int2|
---------------------------------
  ...
---------------------------------
data end
```

ここで、string**は文字列を示す。また int*は整数を示す。例を以下に示す。

```
page|first     |second    |third|
---------------------------------
   1|      jack| mathmatic|   97|
---------------------------------
   2|      pole|  scientic|   83|
---------------------------------
  ...
---------------------------------
data end
```