ファイル操作

古城隆人

2024 年 5 月 16 日

ファイル操作とコマンド引数

1　課題の内容

1.1　課題 1

キーボードから入力された文字列を書き込むプログラムを作成する。要件は以下のとおりである。

- 書き込むファイル名はアプリの起動時に指定する。
- オプションとして-i がある場合があり、これが指定された場合には、書き込むファイルの有無を調べる、もしすでにファイルがあったら上書きしてよいか問い合わせる、その結果、上書きしてはいけない場合には処理を終了する。
- キーボードからエンターキーを押されるまでの文字をファイルに書き込む。
- 上記ののちに終了するか問い合わせ、終了しないならもう一度上記の処理を行う。

1.2　課題 2

シュミレーション結果をファイルに書き込むプログラムを作成する。要件は以下のとおりである。

- 3 桁の数字をランダムに発生させる。
- もし 3 桁がすべて同じ値なら「あたり」、そうでなければ「はずれ」という文字列を発生させる。
- 発生させた数字や文字列をファイルに保存する。
- 最後に期待値と実際の確率をファイルに書き込む。
- 保存ファイル名は main 関数への引数にする。
- シミュレーションする回数も main 関数への引数にする。

2　プログラムリスト

ソースコード 1　main.c

```
1  #define PROGRAM2
2  #ifdef PROGRAM1//課題1
3  #define _CRT_SECURE_NO_WARNINGS
4  #include <stdio.h>
5  #include <stdint.h>
6  #include <stdlib.h>
7  #include <string.h>
8
9  int main(int argc, char* argv[])
10 {
11   FILE* wfp; // 書き込み用ファイルのポインタ変数
12   char input[100] = "y\0";
13   int ch;
```

```
14    if ((argc == 3) && (strcmp(argv[1], "-i") == 0)) {
15      wfp = fopen(argv[2], "r+");
16      if (wfp == NULL)
17      {
18        wfp = fopen(argv[2], "wb");
19      }
20      else {
21        printf("%s上書きしてよいですか?[Y/N]", argv[2]);
22        do
23        {
24        loop:
25          ch = getchar();
26          if (getchar() != '\n') {
27            goto loop;
28          }
29        } while (ch != 'y' && ch != 'Y' && ch != 'n' && ch != 'N');
30        if (ch == 'n' || ch == 'N') {
31          fclose(wfp);
32          return -1;
33        }
34      }
35    }
36    else if ((argc == 2) && (strcmp(argv[1], "-i") != 0)) {
37      if ((wfp = fopen(argv[1], "wb")) == NULL)
38      {
39        printf("Error occurs\n");
40        return -1;
41      }
42    }
43    else {
44      printf("Usage: [-i] file1\n");
45      return -1;
46    }
47    while (strcmp(input, "Y\0") == 0 || strcmp(input, "y\0") == 0)
48    {
49      printf("文字を入力してください\n");
50      scanf("%s", input);
51      fprintf(wfp, "%s\n", input);
52      while (strcmp(input, "Y\0") != 0 && strcmp(input, "y\0") != 0 && strcmp(input, "N\0")
          != 0 && strcmp(input, "n\0") != 0)
53      {
54        printf("入力を続けますか?[Y/N]");
55        scanf("%s", input);
56      }
57    }
58    fclose(wfp);
59    return 0;
60  }
61  #endif // PROGRAM61
```

```
62  #ifdef PROGRAM2 //課題2
63  #define _CRT_SECURE_NO_WARNINGS
64  #include <stdio.h>
65  #include <stdint.h>
66  #include <stdlib.h>
67  #include <time.h>
68
69  int main(int argc, char* argv[])
70  {
71    srand((uint16_t)time(NULL));
72    FILE* rfp; // 読み込み用ファイルのポインタ変数
73    int ch; // ファイルから読み込んだ1バイトのデータ
74    int count = 0;
75    int j = 0;
76    if (argc != 3) {
77      printf("Usage: random file1 count\n");
78      return -1;
79    }
80    if ((rfp = fopen(argv[1], "wb")) == NULL)
81    {
82      printf("Error occurs\n");
83      return -1;
84    }
85    count = atoi(argv[2]);
86    for (int i = 0; i < count; i++)
87    {
88      int ran = rand() % 1000;
89      if ((ran % 111) == 0) {
90        j++;
91        fprintf(rfp, "%3d %s \n", ran, "あたり");
92      }
93      else {
94        fprintf(rfp, "%3d %s \n", ran, "はずれ");
95      }
96    }
97    fprintf(rfp, "試行回数: %d, 当たり回数: %d, 期待値: 1.000000[%%], 確率: %f[%%]",count,j,
        (double)j / (double)count * 100);
98    fclose(rfp);
99    return 0;
100 }
101 #endif // PROGRAM3
```

3　プログラムの説明

define を使用して 2 個のプログラムを一個のプログラムにまとめた。

3.1 プログラム 1 個目

3.1.1 main 関数

main 関数では、表 1 の変数が宣言されています。
また、表 2 の変数を main 関数の引数として受け取ります。
以下プログラムの各行ごとの説明を行う。

- 14 行目の if 文でモードの判別を行っている。
- 15 行目で指定されたファイルが存在するかを確認し、ない場合は NULL となるため次の if 文で true となる。
- 20 行目の else 文に入るときは引数で指定したファイルが存在しているときである。
- 24 行目から 27 行目では goto 文を用いて標準入力のバッファーが空になるまで繰り返す。
- 29 行目の while 文の条件式にて Y か N の入力かを確認している。
- 30 行目でもし、上書きしてはいけない場合は file を閉じてプログラムをステータス-1 で終了する。
- 36 行目で-i オプションが指定されてないとみなし、引数が 1 個であることと-i ではないことを確認している。
- 37 行目で指定されたファイルを読み込むためのポインタを取得している。
- 38 行目から 41 行目でファイルが存在しない場合はエラーを出力してプログラムを終了する。
- 42 行目から 46 行目では引数の入力 error なので、エラーを出力してプログラムを終了する。
- 47 行目から 56 行目では while 文を使用してファイルに文字の入力をしている。

型	変数名	説明
FILE*	wfp	書き込み用のファイルのポインタ変数
char[100]	input	入力された変数を保持する変数
int	ch	標準入力で入力された一文字を保持するための変数

表 1　main 関数で宣言されている変数のリスト

型	変数名	説明
int	argc	メイン関数の引数の数
char*[]	argv	メイン関数の引数の char[] 型の配列の先頭アドレスのポインタ

表 2　main 関数で宣言されている変数のリスト

3.2 プログラム 2 個目

3.2.1 main 関数

main 関数では、表 3 の変数が宣言されています。
また、表 4 の変数を main 関数の引数として受け取ります。
以下プログラムの各行ごとの説明を行う。

- 70 行目で srand 関数を呼び出し、今の時間の timestamp を seed として登録している。
- 75 行目から 78 行目では main 関数の引数が 2 個であることを確認している。引数が 2 個でない場合はエラーを出力してプログラムを終了する。
- 79 行目から 83 行目では引数で指定されたファイルを読み込むためのポインタを取得している。
- 84 行目では count 変数に引数で指定された回数を代入している。atoi 関数は文字列を数値に変換する関数である。
- 85 行目から 95 行目では for 文を使用して count 回数だけ乱数を生成している。乱数が 3 桁とも同じ数字である場合は j 変数をインクリメントする。また、fprintf 関数を使用してファイルに乱数と結果を書き込んでいる。
- 96 行目では試行回数、あたり回数、期待値、確率をファイルに書き込んでいる。
- 99 行目ではファイルを閉じてプログラムを終了する。

型	変数名	説明
FILE*	rfp	書き込み用のファイルのポインタ変数
int	ch	ファイルから読み込んだ文字を保持する変数
int	count	乱数を生成する試行回数
int	j	乱数が 3 桁ともそろった回数を保存する変数

表 3　main 関数で宣言されている変数のリスト

型	変数名	説明
int	argc	メイン関数の引数の数
char*[]	argv	メイン関数の引数の char[] 型の配列の先頭アドレスのポインタ

表 4　main 関数で宣言されている変数のリスト

4　結果の説明

4.1　プログラム 1 個目

　ファイルが存在しない場合の実行結果を図 1 に示す。
ファイルが存在する場合の実行結果を図 2 に示す。
また、ファイルの出力結果を表 5 に示す。abc1.txt ではファイルが存在しないとき、abc2.txt ではファイルが存在するときの実行結果である。どちらの場合も同じ引数で実行しているため、場合分けして実行した後にファイル名を変更している。

ファイル名	出力結果
abc1.txt	test aiueo
abc2.txt	aiueo ueo

表 5　ファイルの出力結果
(abc のときがファイルが存在しないときで、abc2 のときがファイルが存在するとき)

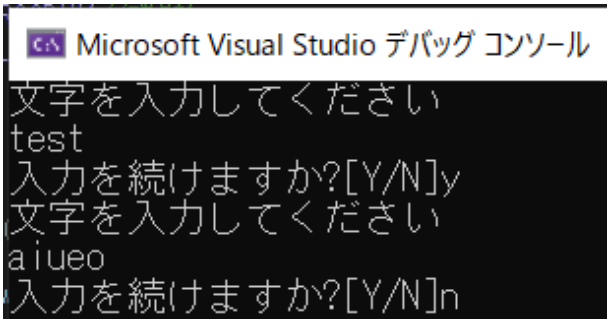

図 1　ファイルが存在しないときの実行結果

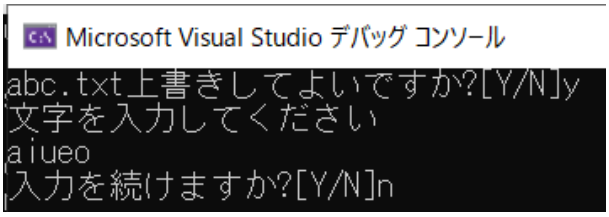

図 2　ファイルが存在するときの実行結果

4.2　プログラム 2 個目

ファイルの出力結果を図 3 に示す。

図 3　ファイルの出力結果

5　考察

5.1　プログラム 1 個目

　ファイルが存在しない場合と存在する場合での実行結果を比較すると、上書きした時に上の行だけでなく、下の行の文字も上書きされていた。ここから考えられることとしては、ファイルを読み込んだらその値がメモリに格納され fprintf でそのメモリに干渉して上書きしていると読み取れる。なので fprintf や fputc を行っただけではファイルにメモリの値が書き込まれるわけではないと考えられる。

5.2　プログラム 2 個目

　乱数を生成してファイルに書き込むプログラムを作成出来たと思う。
生成した乱数が 3 桁とも同じ確率が毎回ぶれていたけど、回数を増やせば確率が 1% の期待値に近づくと思った。
そこで追加実験として、試行回数を 123456 回にして実行したところ、期待値と実際の確率が近い値になった。
実行結果を図 4 に示す。

図 4　試行回数を 123456 回にしたときの実行結果