

課題 No.1 の解答例

低水準入出力を直接使用するファイルコピープログラムを作成した。

実行例

実行例の内容は、リストの右側に書き込んだコメントを参考に確認すること。

リスト 1: 実行例（動作テスト）

```
1 % cc -o mycp2 mycp2.c <-- エラーも警告も出ない
2 % echo aaa > a.txt <-- 内容が"aaa\n"のファイル作成
3 % ./mycp2 <-- コマンド行引数がない場合
4 Usage: ./mycp2 srcfile dstfile
5 % ./mycp2 a.txt <-- コマンド行引数が不足の場合
6 Usage: ./mycp2 srcfile dstfile
7 % ./mycp2 a.txt b.txt c.txt <-- コマンド行引数が過剰な場合
8 Usage: ./mycp2 srcfile dstfile
9 % ./mycp2 z.txt a.txt <-- コピー元が存在しない場合
10 z.txt: No such file or directory
11 % ./mycp2 a.txt /a.txt <-- コピー先が書き込み禁止の場合
12 /a.txt: Permission denied
13 % ./mycp2 a.txt b.txt <-- b.txt にコピーしてみる
14 % cat b.txt <-- b.txt の内容を確認
15 aaa
16 % echo ccc > c.txt <-- 内容が"ccc\n"のファイル作成
17 % ./mycp2 c.txt b.txt <-- b.txt に上書きしてみる
18 % cat b.txt <-- b.txt の内容を再確認
19 ccc
20 % cmp c.txt b.txt <-- b.txt の内容を再々確認
21 % dd if=/dev/urandom of=srcfile bs=1024 count=10 <-- 10KiBの長いファイルを作る
22 10+0 records in
23 10+0 records out
24 10240 bytes transferred in 0.001695 secs (6041591 bytes/sec)
25 % rm destfile
26 rm: destfile: No such file or directory <-- destfile が存在しない場合
27 % ./mycp2 srcfile destfile
28 % cmp srcfile destfile <-- 正しくコピーできている
29 % dd if=/dev/urandom of=srcfile bs=1023 count=10 <-- 10KiBより少し短いファイル
30 10+0 records in
31 10+0 records out
32 10230 bytes transferred in 0.003218 secs (3179057 bytes/sec)
33 % ./mycp2 srcfile destfile <-- destfile が短くなる場合
34 % cmp srcfile destfile <-- 正しくコピーできている
```

ソースプログラム

7行 バッファサイズを#define で定義している。

16行 コマンド行引数の数を確認している。確認しないと後の処理で誤動作を引き起こす。

22 行 読み込み用のオープンには O_RDONLY フラグを用いる。

29 行 書き込み用のオープンには O_WRONLY, O_CREATE, O_TRUNC フラグを用いるのが適切である。ファイルが短くなる場合に O_TRUNC が必要である。ファイルを新規作成する場合は保護モードも元ファイルと同じにすべきだが, rw-r--r-- に固定とした。

37 行 読み込んだデータのバイト数だけ書き込むように len を用いる。

リスト 2: 低水準 I/O 版の mycp(mycp2.c)

```
1 #include <stdio.h> // perror のため
2 #include <stdlib.h> // exit のため
3 #include <fcntl.h> // open のため
4 #include <unistd.h> // read, write, close のため
5
6 // #define BSIZ 1 // !!バッファサイズ: 変化させ性能を調べる!!
7 #define BSIZ 1024 // !!バッファサイズ: 変化させ性能を調べる!!
8
9 int main(int argc, char *argv[]) {
10     int fd1; // コピー元用のFD
11     int fd2; // コピー先のFD
12     ssize_t len; // 実際に読んだバイト数
13     char buf[BSIZ]; // バッファ
14
15     // ユーザの使い方エラーのチェック
16     if (argc != 3) {
17         fprintf(stderr, "Usage : %s srcfile dstfile\n", argv[0]);
18         return 1; // exit(1); と同じ
19     }
20
21     // 読み込み用にファイルオープン
22     fd1 = open(argv[1], O_RDONLY);
23     if (fd1 < 0) { // コピー元のオープンエラーのチェック
24         perror(argv[1]); // エラーメッセージ表示
25         return 1; // exit(1); と同じ
26     }
27
28     // 書き込み用にファイルオープン
29     fd2 = open(argv[2], O_WRONLY|O_CREAT|O_TRUNC, 0644);
30     if (fd2 < 0) { // コピー先のオープンエラーのチェック
31         perror(argv[2]); // エラーメッセージ表示
32         return 1; // exit(1); と同じ
33     }
34
35     // ファイルの書き写し
36     while ((len = read(fd1, buf, BSIZ)) > 0) {
37         write(fd2, buf, len);
38     }
39
40     close(fd1);
41     close(fd2);
42     return 0; // 正常終了
43 }
```