

# ネットワークプログラミング

## 第3回演習レポート

222C1021 今村優希

2024年6月25日

## 演習 1

左側のプログラムを実行させたときの結果は、図??である。

```
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ./ensyuu
2
2! = 2
3
3! = 6
6
6! = 720
11
11! = 39916800
```

図 1 実行結果

また、右側のプログラムを実行させた。”input1.dat”を読み込ませて、”output1.dat”に書き込ませ、出力させたものが図 2 である。

```
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ cat output1.dat
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
```

図 2 output1.dat の出力

## 演習 2

作成したのは、ソースコード 1 である。このプログラムを実行した結果がである。

ソースコード 1 演習 2 のプログラム

```
#include <stdio.h>
#include <stdlib.h>

// 構造体の作成
struct student{
    char id[10];
    int score;
};

int main(void){
    struct student seito[10];
    int i = 0;
```

```

FILE *fin, *fout;
fin = fopen("input2.dat", "r");    //input2.datを読み込み専用で開く
fout = fopen("output2.dat", "w");  //output2.datを書き込み専用で開く

int allscore = 0;                  //合計点用
double ave = 0;                   //平均点用

if(fin == NULL){
    printf("No file\n");
    exit(1);
} else {
    while(feof(fin) == 0){
        fscanf(fin,"%s %d\n",seito[i].id, &seito[i].score); // ファイル
        入出力
        allscore = allscore + seito[i].score;
        i++;
    }
    ave = allscore / i;
    fprintf(fout,"学生数 = %d\n平均点 = %f\n",i, ave);
}

fclose(fin);                      //fileクローズ
fclose(fout);

return 0;
}

```

```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ cat output2.dat
学生数 = 5
平均点 = 83.000000

```

図3 ソースコード1の実行結果

### 演習 3

作成したプログラムは、ソースコード2である。”output3.dat”に出力し、それを出力させたものが図4である。

ソースコード2 演習3のプログラム

```

#include <stdio.h>
#include <string.h>    // strlen() に必要
#include <fcntl.h>     // open() に必要
#include <unistd.h>
// read(), write(), close() に必要

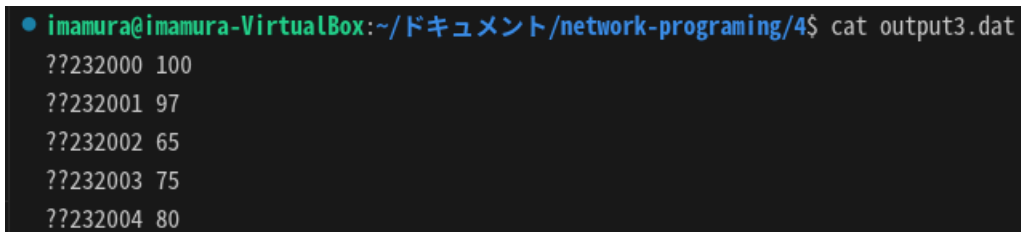
```

```

int main(){
    char buf[13];
    int fin, fout;
    fin = open("input2.dat", O_RDONLY);
    fout = open("output3.dat", O_WRONLY);

    if(fin < 0){
        exit(1);
    }
    // 入力ファイル指定
    while(read(fin, buf, sizeof(buf)) != 0){
        write(fout, buf, strlen(buf)); //書き込み
        memset(buf, '\0', sizeof(buf)); //繰り返し用でbufを初期化
    }
    write(fout, "\n", strlen("\n"));
    close(fin);
    close(fout);
}

```



```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ cat output3.dat
??232000 100
??232001 97
??232002 65
??232003 75
??232004 80

```

図4 "output3.dat"の出力結果

## 演習 4

プロセス生成を行った場合の実行結果が5である。また、リダイレクションを行った場合の実行結果が6である。リダイレクションは、temp.txt に出力が書き込まれている。

```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ gcc -o ensyuu 3-12.c
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ./ensyuu
合計 80
-rw-rw-r-- 1 imamura imamura 333 6月 25 18:36 3-1.c
-rw-rw-r-- 1 imamura imamura 623 6月 25 18:36 3-12.c
-rw-rw-r-- 1 imamura imamura 1087 6月 25 18:36 3-15.c
-rw-rw-r-- 1 imamura imamura 495 6月 25 18:36 3-2.c
-rw-rw-r-- 1 imamura imamura 467 6月 25 18:36 3-4.c
-rw-rw-r-- 1 imamura imamura 322 6月 25 18:36 3-5.c
-rw-rw-r-- 1 imamura imamura 946 6月 25 18:36 e3-2.c
-rw-rw-r-- 1 imamura imamura 583 6月 25 18:36 e3-3.c
-rw-rw-r-- 1 imamura imamura 1435 6月 25 18:36 e3-5.c
-rwxrwxr-x 1 imamura imamura 16128 6月 25 19:00 ensyuu
-rw-rw-r-- 1 imamura imamura 12 6月 25 18:36 input1.dat
-rw-rw-r-- 1 imamura imamura 60 6月 25 18:36 input2.dat
-rw-rw-r-- 1 imamura imamura 47 6月 25 18:57 output1.dat
-rw-rw-r-- 1 imamura imamura 36 6月 25 18:36 output2.dat
-rw-rw-r-- 1 imamura imamura 61 6月 25 18:36 output3.dat
-rw-rw-r-- 1 imamura imamura 123 6月 25 18:36 temp.txt
-rw-rw-r-- 1 imamura imamura 1002 6月 25 18:36 test.txt
child process 8859 is finished

```

図5 プロセス生成の実行結果

```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ gcc -o ensyuu 3-15.c
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ./ensyuu
child process 8944 is finished
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ cat temp.txt
PID TTY          TIME CMD
6707 pts/0        00:00:00 bash
8943 pts/0        00:00:00 ensyuu
8944 pts/0        00:00:00 ps

```

図6 リダイレクションの実行結果

## 演習 5

まずは、p.17 のコマンドライン実行を行った。行った結果が、7である。

また、親子プロセス間通信を実現するためのプログラムは、ソースコード 3 である。このプログラムを実行させた時に結果が、8 である。

ソースコード 3 親子プロセス間を実現するプログラム

```

#include <stdio.h>          // fprintf() の利用 に 必要
#include <stdlib.h>
#include <unistd.h>          // fork(), exec()
#include <sys/types.h>       // fork(), wait() に 必要

```

```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ls -l > test.txt ; grep c < test.txt
-rw-rw-r-- 1 imamura imamura  333  6月 25 18:36 3-1.c
-rw-rw-r-- 1 imamura imamura  623  6月 25 18:36 3-12.c
-rw-rw-r-- 1 imamura imamura 1087  6月 25 18:36 3-15.c
-rw-rw-r-- 1 imamura imamura  495  6月 25 18:36 3-2.c
-rw-rw-r-- 1 imamura imamura  467  6月 25 18:36 3-4.c
-rw-rw-r-- 1 imamura imamura  322  6月 25 18:36 3-5.c
-rw-rw-r-- 1 imamura imamura  946  6月 25 18:36 e3-2.c
-rw-rw-r-- 1 imamura imamura  583  6月 25 18:36 e3-3.c
-rw-rw-r-- 1 imamura imamura 1435  6月 25 18:36 e3-5.c
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ls -l | grep put
-rw-rw-r-- 1 imamura imamura   12  6月 25 18:36 input1.dat
-rw-rw-r-- 1 imamura imamura   60  6月 25 18:36 input2.dat
-rw-rw-r-- 1 imamura imamura   47  6月 25 18:57 output1.dat
-rw-rw-r-- 1 imamura imamura   36  6月 25 18:36 output2.dat
-rw-rw-r-- 1 imamura imamura   61  6月 25 18:36 output3.dat

```

図7 コマンドラインでの実行結果

```

#include <sys/wait.h>    // wait() に必要
#include <fcntl.h>        // creat(),close() に必要

int main()
{
    int fd[2];
    pid_t pwait;          // wait() からの戻り値
    int status;           // 子プロセスの終了状態

    if(pipe(fd) == -1){ // 単方向パイプの作成
        perror("pipe");
        exit(1);
    }

    if(fork() == 0) {
        close(fd[0]);     // 不要なパイプを閉じる

        close(1);         // 標準出力を閉じる
        dup(fd[1]);        // パイプfd[1]をコピー

        close(fd[1]);     // 不要なパイプfd[1]を閉じる
        execl("/bin/ls", "ls", "-l", NULL); //

        fprintf(stderr, "cannot ls command\n");
        exit(1);          // execl() 失敗時は if 文を抜ける
    }
}

```

```

else{                                // 親プロセス側の処理
    pwait = wait(&status); //子プロセス待ち
    printf("child process %d is finished\n", pwait);

    close(fd[1]); //不要なパイプを閉じる
    close(0);    //標準入力を閉じる
    dup(fd[0]);  //パイプfd[0]をコピー

    close(fd[0]); //不要なパイプfd[0]を閉じる
    execl("/bin/grep","grep","put", NULL);

    fprintf(stderr,"cannot grep command\n");
    exit(1);
}
}

```

```

● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ gcc -o ensyuu e3-5.c
● imamura@imamura-VirtualBox:~/ドキュメント/network-programing/4$ ./ensyuu
child process 9274 is finished
-rw-rw-r-- 1 imamura imamura 12  6月 25 18:36 input1.dat
-rw-rw-r-- 1 imamura imamura 60  6月 25 18:36 input2.dat
-rw-rw-r-- 1 imamura imamura 47  6月 25 18:57 output1.dat
-rw-rw-r-- 1 imamura imamura 36  6月 25 18:36 output2.dat
-rw-rw-r-- 1 imamura imamura 61  6月 25 18:36 output3.dat

```

図8 ソースコード3の実行結果