

ネットワークプログラミング

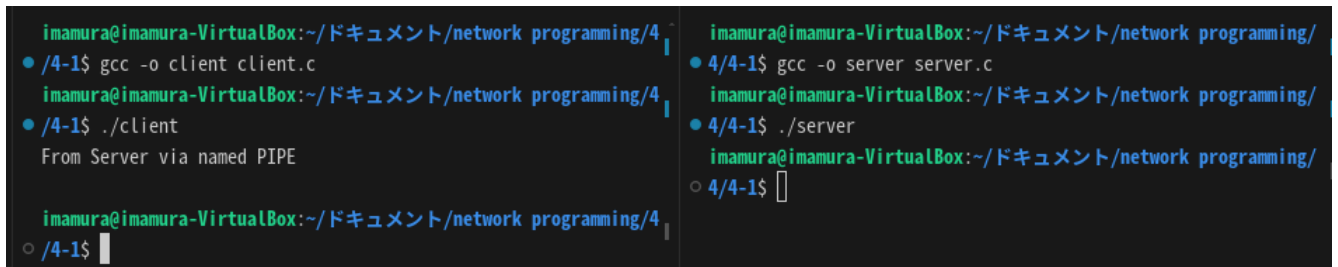
第 6 回演習レポート

222C1021 今村優希

2024 年 7 月 1 日

演習 1

作成した server.c と client.c を実行した結果が図 1 である． server 側で設定した文字列を client 側の出力できるのを確認できた．



```
imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
● /4-1$ gcc -o client client.c
imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
● /4-1$ ./client
From Server via named PIPE
imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
○ /4-1$

imamura@imamura-VirtualBox:~/ドキュメント/network programming/
● 4/4-1$ gcc -o server server.c
imamura@imamura-VirtualBox:~/ドキュメント/network programming/
● 4/4-1$ ./server
imamura@imamura-VirtualBox:~/ドキュメント/network programming/
○ 4/4-1$
```

図 1 server.c と client.c の実行結果

演習 2

作成した server2.c と client2.c はソースコード 1,2 である。このプログラムをコンパイルし、実行した結果が図 2 である。左側が client.c を、右側を server.c を実行した結果を表している。server 側で入力した文字列を client 側で同時に出力することができた。また、imamura と入力することで終了させることもできた。

なお、server2.c における `scanf("%s", buf);` に対して `scanf("%s\n", buf);` のように、`\` を加えると入力したものが遅れて出力される現象が確認できた。

ソースコード 1 server2.c

```
/*222C1021 今村優希*/
/*server2.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h> // strlen()
#include <fcntl.h> // open(), creat()
#include <unistd.h> // read(), write(), close()
#define PIPE "/tmp/mypipe" // 1.
// 名前付きパイプの指定
int main(){
    char buf[80] = "From Server via named PIPE";
    char test[10] = "imamura";
    int fd;
    int i = 0;
    fd = open(PIPE, O_WRONLY); // 2.
    while(strcmp(buf, test) != 0){

        if (fd == -1){
            fprintf(stderr, "PIPE does not exist! \n");
            exit(1);
        }
        write(fd, buf, strlen(buf)); // 3.
        scanf("%s", buf);
    }
    close(fd);
}
```

ソースコード 2 client2.c

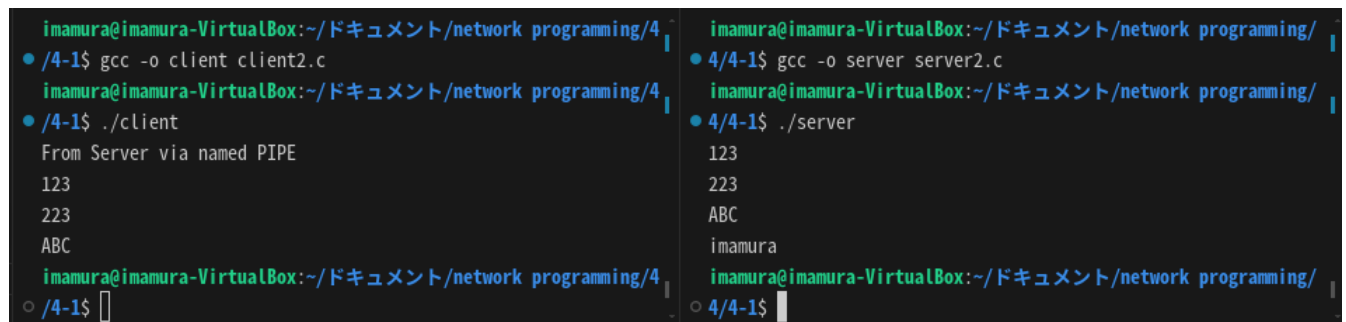
```
/*222C1021 今村優希*/
/*client2.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h> // strlen()
```

```

#include <fcntl.h> // open(), creat()
#include <unistd.h> // read(), write(), close()
#define PIPE "/tmp/mypipe" // 1.
// 名前付きパイプの指定
int main(){
    char buf[80]; // = "From Server via named PIPE";
    int fd;
    fd = open(PIPE, O_RDONLY); // 2.
    if (fd == -1){
        fprintf(stderr, "PIPE does not exist! \n");
        exit(1);
    }
    while(read(fd, buf, sizeof(buf)) != 0){
        printf("%s\n", buf);    // 書き込み
        memset(buf, '\0', sizeof(buf)); // 繰り返し用でbufを初期化
    }
    close(fd); // 4.
}

```



```

imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
● /4-1$ gcc -o client client2.c
imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
● /4-1$ ./client
From Server via named PIPE
123
223
ABC
imamura@imamura-VirtualBox:~/ドキュメント/network programming/4
○ /4-1$

imamura@imamura-VirtualBox:~/ドキュメント/network programming/
● 4/4-1$ gcc -o server server2.c
imamura@imamura-VirtualBox:~/ドキュメント/network programming/
● 4/4-1$ ./server
123
223
ABC
imamura
imamura@imamura-VirtualBox:~/ドキュメント/network programming/
○ 4/4-1$

```

図2 server2.c と client2.c の実行結果

演習 3

作成した server3.c と client3.c はソースコード 3,4 である。このプログラムをコンパイルし、実行した結果が図 3 である。左側が client3.c を、右側を server3.c を実行した結果である。server 側では client 側で設定した IP アドレス等を出力することができたし、client 側では server 側で設定した IP アドレス等を出力することができた。

ソースコード 3 server3.c

```
/*222C1021 今村優希*/
/*server3.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int sockfd_s;
    struct sockaddr_in address_s;
    char buf[80] = "\0";

    //INETドメイン、ストリームソケットを利用
    sockfd_s = socket(AF_INET, SOCK_STREAM, 0);

    //サーバソケット=(IP: 10.0.2.15, ポート:5000)に設定
    address_s.sin_family = AF_INET;
    address_s.sin_addr.s_addr = inet_addr("10.0.2.15");
    address_s.sin_port = htons(5000);

    bind(sockfd_s, (struct sockaddr *)&address_s, sizeof(address_s));

    //要求受付の準備
    listen(sockfd_s, 5);
    printf("server waits\n");

    //接続要求の許可と情報確認
    struct sockaddr_in address_c;
    unsigned int length_c = sizeof(address_c);
```

```

    int sockfd_c = accept(sockfd_s, (struct sockaddr *)& address_c, &
        length_c);

    printf("\n * request from client IP: %s, port %d\n", inet_ntoa(address_c
        .sin_addr), ntohs(address_c.sin_port));

    //送受信操作
    memset(buf, '\0', sizeof(buf));
    read(sockfd_c, buf, sizeof(buf));
    printf("\n* message from client: %s\n", buf);

    strcpy(buf, "From Server via socket");
    write(sockfd_c, buf, strlen(buf));

    //ソケットの除去
    close(sockfd_c);
    close(sockfd_s);
}

```

ソースコード 4 client3.c

```

/*222C1021 今村優希*/
/*client3.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int sockfd;
    struct sockaddr_in address;
    char buf[80] = "\0";

    //INETドメイン、ストリームソケットを利用
    sockfd = socket(AF_INET, SOCK_STREAM, 0);

    //サーバー情報を設定
    address.sin_family = AF_INET;

```

```

address.sin_addr.s_addr = inet_addr("10.0.2.15");
address.sin_port = htons(5000);

//クライアントからの接続要求とサーバ情報を確認
int res = connect(sockfd, (struct sockaddr *)&address, sizeof(address));

if(res == -1){
    perror("error\n");
    exit(1);
}
printf("\n * server IP: %s, port: %d\n", inet_ntoa(address.sin_addr),
        ntohs(address.sin_port));

strcpy(buf, "client will connect to server");
write(sockfd, buf, strlen(buf));

//サーバからのデータ受信
memset(buf, '\0', sizeof(buf)); // buf[] 読み込み前に初期化
read(sockfd, buf, sizeof(buf));
printf("\n * message from server : %s \n", buf);

//ソケットの除去
close(sockfd);
}

```

<pre> imamura@imamura-VirtualBox:~/ドキュメント/network programming/4 ● /4-2\$ gcc -o client client3.c imamura@imamura-VirtualBox:~/ドキュメント/network programming/4 ● /4-2\$./client * server IP: 10.0.2.15, port: 5000 * message from server : From Server via socket imamura@imamura-VirtualBox:~/ドキュメント/network programming/4 ○ /4-2\$ </pre>	<pre> imamura@imamura-VirtualBox:~/ドキュメント/network programming/ ● 4/4-2\$ gcc -o server server3.c imamura@imamura-VirtualBox:~/ドキュメント/network programming/ ● 4/4-2\$./server server waits * request from client IP: 10.0.2.15, port 56852 * message from client: client will connect to server imamura@imamura-VirtualBox:~/ドキュメント/network programming/ ○ 4/4-2\$ </pre>
--	--

図3 server3.c と client3.c の実行結果

演習 4

作成した server4.c と client4.c はソースコード 6,5 である。このプログラムをコンパイルし、実行した結果が図 4 である。左側と真ん中が client4.c を、右側を server4.c を実行した結果である。左側と真ん中は同時に多少時間を開けて実行したとしても、0 から 10 まで出力されるのが確認できた。

ソースコード 5 server4.c

```
/*222C1021 今村優希*/
/*server4.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/wait.h>

int main(){
    int sockfd_s, sockfd_c, i, n;
    struct sockaddr_in address_s;
    char buf[80] = "\0";

    // INETドメイン、ストリームソケットを利用
    sockfd_s = socket(AF_INET, SOCK_STREAM, 0);

    // サーバソケット=(IP: 10.0.2.15, ポート:5000)に設定
    address_s.sin_family = AF_INET;
    address_s.sin_addr.s_addr = inet_addr("10.0.2.15");
    address_s.sin_port = htons(5000);

    bind(sockfd_s, (struct sockaddr *)&address_s, sizeof(address_s));

    // 要求受付の準備
    listen(sockfd_s, 5);
    printf("server waits\n");

    // 繰り返し
    while(1){
        // 接続要求の許可と情報確認
```



```

    struct sockaddr_in address_c;
    unsigned int length_c = sizeof(address_c);
    sockfd_c = accept(sockfd_s, (struct sockaddr *)& address_c, &
        length_c);

    //fork文、子プロセス作成
    if(fork() != 0) {    // 子プロセス側の処理、送受信操作
        close(sockfd_s);
        for(i = 0; i <= 10; i++){
            memset(buf, '\0', sizeof(buf));
            n = snprintf(buf, sizeof(i), "%d", i);
            write(sockfd_c, buf, strlen(buf));
            sleep(1);
        }
        close(sockfd_c);
        exit(0);
    }
    else{                // 親プロセス側の処理、要求受付
        close(sockfd_c);
    }
    close(sockfd_c);
}    //繰り返し終了

//ソケットの除去
close(sockfd_s);
}

```

ソースコード 6 client4.c

```

/*222C1021 今村優希*/
/*client4.c*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>

int main(){
    int sockfd, i;

```

```

struct sockaddr_in address;
char buf[80] = "\0";

// INETドメイン、ストリームソケットを利用
sockfd = socket(AF_INET, SOCK_STREAM, 0);

// サーバ情報を設定
address.sin_family = AF_INET;
address.sin_addr.s_addr = inet_addr("10.0.2.15");
address.sin_port = htons(5000);

// クライアントからの接続要求とサーバ情報を確認
int res = connect(sockfd, (struct sockaddr *)&address, sizeof(address));

if(res == -1){
    perror("error\n");
    exit(1);
}

while(i != 10){ // i=10になったら読み込み終了
    // サーバからのデータ受信
    memset(buf, '\0', sizeof(buf)); // buf[] 読み込み前に初期化
    read(sockfd, buf, sizeof(buf));
    i = atoi(buf);
    printf("\n * message from server : %d \n", i);
}
// ソケットの除去
close(sockfd);
}

```

```
imamura@imamura-VirtualBox:~/ドキュメント/network program$ ./client
* message from server : 0
* message from server : 1
* message from server : 2
* message from server : 3
* message from server : 4
* message from server : 5
* message from server : 6
* message from server : 7
* message from server : 8
* message from server : 9
* message from server : 10
imamura@imamura-VirtualBox:~/ドキュメント/network program$

imamura@imamura-VirtualBox:~/ドキュメント/network program$ ./client
* message from server : 0
* message from server : 1
* message from server : 2
* message from server : 3
* message from server : 4
* message from server : 5
* message from server : 6
* message from server : 7
* message from server : 8
* message from server : 9
* message from server : 10
imamura@imamura-VirtualBox:~/ドキュメント/network program$

imamura@imamura-VirtualBox:~/ドキュメント/network program$ gcc -o server server4.c
imamura@imamura-VirtualBox:~/ドキュメント/network program$ ./server
server waits
imamura@imamura-VirtualBox:~/ドキュメント/network program$ ^C
imamura@imamura-VirtualBox:~/ドキュメント/network program$
```

図 4 server4.c と client4.c の実行結果

備考

今回作成したプログラムを github 上に公開した.

https://github.com/tanusai646/network_program/tree/main/4