EXP-1::::

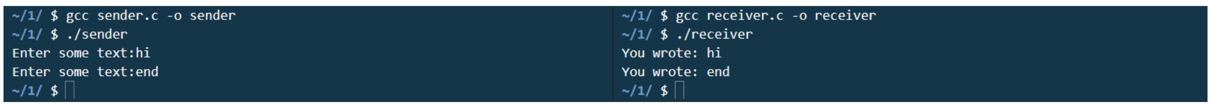
Sender-

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
#define MAX_TEXT 512
struct my_msg_st {
long int my_msg_type;
char some_text[MAX_TEXT];
};
int main()
{
int running = 1;
struct my_msg_st some_data;
int msgid;
char buffer[BUFSIZ];
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
printf("Enter some text:");
fgets(buffer, BUFSIZ, stdin);
some_data.my_msg_type = 1;
strcpy(some_data.some_text, buffer);
if (msgsnd(msgid, (void *)&some_data, MAX_TEXT, 0) == -1) {
fprintf(stderr, "msgsnd failed\n");
exit(EXIT_FAILURE);
}
if (strncmp(buffer, "end", 3) == 0) {
running = 0;
}
}
exit(EXIT_SUCCESS);
}
```

Receiver-

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/msg.h>
```

```
struct my_msg_st {
long int my_msg_type;
char some_text[BUFSIZ];
};
int main()
{
int running = 1;
int msgid;
struct my_msg_st some_data;
long int msg_to_receive = 0;
msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
if (msgid == -1) {
fprintf(stderr, "msgget failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
while(running) {
if (msgrcv(msgid, (void *)&some_data, BUFSIZ, msg_to_receive, 0) == -1) {
fprintf(stderr, "msgrcv failed with error: %d\n", errno);
exit(EXIT_FAILURE);
}
printf("You wrote: %s", some_data.some_text);
if (strncmp(some_dataw.some_text, "end", 3) == 0) {
running = 0;
}
}
if (msgctl(msgid, IPC_RMID, 0) == -1) {
fprintf(stderr, "msgctl(IPC_RMID) failed\n");
exit(EXIT_FAILURE);
}
exit(EXIT_SUCCESS);
}
```



```
~/1/ $ gcc sender.c -o sender
~/1/ $ ./sender
Enter some text:hi
Enter some text:end
~/1/ $
```

```
~/1/ $ gcc receiver.c -o receiver
~/1/ $ ./receiver
You wrote: hi
You wrote: end
~/1/ $
```

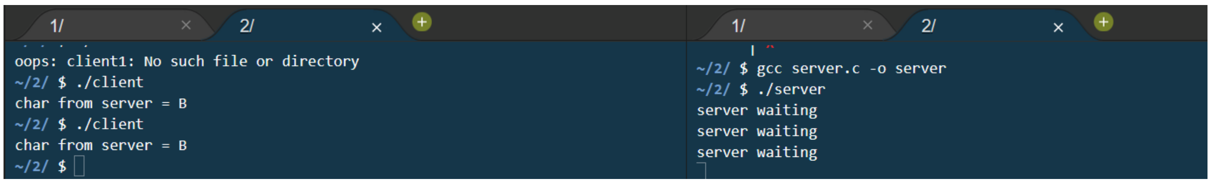
EXP-2:::

Client:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <sys/un.h>
#include <unistd.h>
#include <stdlib.h>
int main()
{ int sockfd;
int len; struct sockaddr_un address;
int result;
char ch = 'A';
sockfd =socket(AF_UNIX, SOCK_STREAM, 0);
address.sun_family = AF_UNIX;
strcpy(address.sun_path, "server_socket");
len = sizeof(address);
result = connect(sockfd, (struct sockaddr *)&address, len);
if(result == -1) { perror("oops: client1");
exit(1);
}
write(sockfd, &ch, 1);
read(sockfd, &ch, 1);
printf("char from server = %c\n", ch);
close(sockfd);
exit(0);
}
```

Server:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <sys/un.h>
#include <unistd.h>
#include <stdlib.h>
int main()
{ int server_sockfd, client_sockfd;
int server_len, client_len;
struct sockaddr_un server_address;
struct sockaddr_un client_address;
unlink("server_socket");
server_sockfd = socket(AF_UNIX, SOCK_STREAM, 0);
server_address.sun_family = AF_UNIX;
strcpy(server_address.sun_path, "server_socket");
server_len = sizeof(server_address);
bind(server_sockfd, (struct sockaddr *)&server_address, server_len);
listen(server_sockfd, 5);
while(1) { char ch;
printf("server waiting\n");
client_len = sizeof(client_address);
client_sockfd = accept(server_sockfd,(struct sockaddr *)&client_address, &client_len);
read(client_sockfd, &ch, 1);
ch++;
write(client_sockfd, &ch, 1);
close(client_sockfd);
}
}
```



```
oops: client1: No such file or directory
~/2/ $ ./client
char from server = B
~/2/ $ ./client
char from server = B
~/2/ $
```

```
~/2/ $ gcc server.c -o server
~/2/ $ ./server
server waiting
server waiting
server waiting
```

EXP-3:::

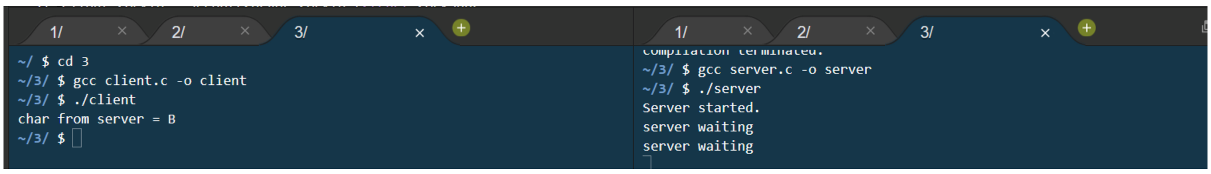
Client

```
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
int main()
{
int sockfd;
int len;
struct sockaddr_in address;
int result;
char ch = 'A';
sockfd = socket(AF_INET, SOCK_STREAM, 0);
address.sin_family = AF_INET;
address.sin_addr.s_addr = inet_addr("127.0.0.1");
address.sin_port = 9734;
len = sizeof(address);
result = connect(sockfd, (struct sockaddr *)&address, len);
if(result == -1) {
perror("oops: client1");
exit(1);
}
write(sockfd, &ch, 1);
read(sockfd, &ch, 1);
printf("char from server = %c\n", ch);
exit(0);
}
```

Server

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

```
#include <string.h>
int main(){
int client_sockfd, server_sockfd;
int server_len, client_len;
struct sockaddr_in server_address;
struct sockaddr_in client_address;
server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
server_address.sin_family = AF_INET;
server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
server_address.sin_port = 9734;
server_len = sizeof(server_address);
bind(server_sockfd, (struct sockaddr *)&server_address,
server_len);
listen(server_sockfd, 5);
printf("Server started.\n");
while(1) {
char ch;
printf("server waiting\n");
client_len = sizeof(client_address);
client_sockfd = accept(server_sockfd,(struct sockaddr
*)&client_address, &client_len);
read(client_sockfd, &ch, 1);
ch++;
write(client_sockfd, &ch, 1);
}
}
```



```
~/ $ cd 3
~/3/ $ gcc client.c -o client
~/3/ $ ./client
char from server = B
~/3/ $
```

```
compilation terminated.
~/3/ $ gcc server.c -o server
~/3/ $ ./server
Server started.
server waiting
server waiting
```

EXP-3b:::

Client:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
int main()
{
  int sockfd;
```

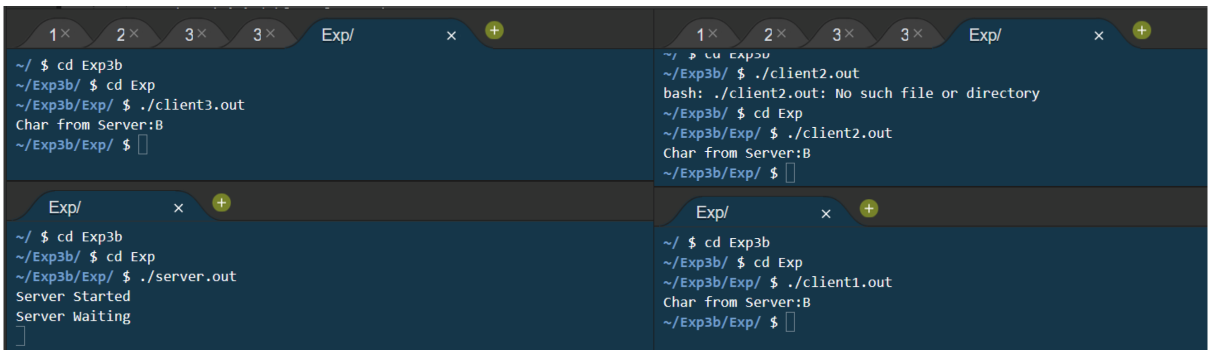
```
    int len;
    struct sockaddr_in address;
    int result;
    char ch = 'A';
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = inet_addr("127.0.0.1");
    address.sin_port = 9734;
    len = sizeof(address);
    result = connect(sockfd, (struct sockaddr *)&address, len);
    if (result == -1)
    {
        perror("oops:client1");
        exit(1);
    }
    write(sockfd, &ch, 1);
    read(sockfd, &ch, 1);
    printf("Char from Server:%c\n", ch);
    close(sockfd);
    return 0;
}
```

Server:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <pthread.h>
#define MAXCLIENT 3

void *server(void *arg)
{
    char ch;
    int client_sockfd, client_len;
    struct sockaddr_in client_address;
    int socket = *((int *)arg);
    client_len = sizeof(client_address);
    client_sockfd = accept(socket, (struct sockaddr *)&client_address, &client_len);
    read(client_sockfd, &ch, 1);
    ch++;
    write(client_sockfd, &ch, 1);
    return NULL;
}
```

```
int main()
{
   int server_sockfd;
   int server_len, temp;
   struct sockaddr_in server_address;
   pthread_t th[MAXCLIENT];
   server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
   server_address.sin_family = AF_INET;
   server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
   server_address.sin_port = 9734;
   server_len = sizeof(server_address);
   bind(server_sockfd, (struct sockaddr *)&server_address, server_len);
   listen(server_sockfd, MAXCLIENT);
   printf("Server Started\n");
   while (1)
   {
      temp = 0;
      while (temp < MAXCLIENT)
      {
         int *pserver = malloc(sizeof(int));
         *pserver = server_sockfd;
         pthread_create(&th[temp], NULL, server, (void *)pserver);
         temp++;
      }
      temp = 0;
      while (temp < MAXCLIENT)
      {
         printf("Server Waiting\n");
         pthread_join(th[temp], NULL);
         temp++;
      }
   }
   return 0;
}
```



```
~/ $ cd Exp3b
~/Exp3b/ $ cd Exp
~/Exp3b/Exp/ $ ./client3.out
Char from Server:B
~/Exp3b/Exp/ $
```

```
~/ $ cd Exp3b
~/Exp3b/ $ ./client2.out
bash: ./client2.out: No such file or directory
~/Exp3b/ $ cd Exp
~/Exp3b/Exp/ $ ./client2.out
Char from Server:B
~/Exp3b/Exp/ $
```

```
~/ $ cd Exp3b
~/Exp3b/ $ cd Exp
~/Exp3b/Exp/ $ ./server.out
Server Started
Server Waiting
```

```
~/ $ cd Exp3b
~/Exp3b/ $ cd Exp
~/Exp3b/Exp/ $ ./client1.out
Char from Server:B
~/Exp3b/Exp/ $
```

EXP5:::

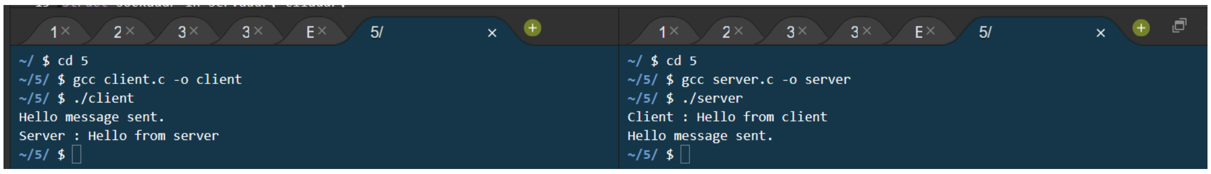
Client

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#define PORT 8080
#define MAXLINE 1024
int main() {
int sockfd;
char buffer[MAXLINE];
char *hello = "Hello from client";
struct sockaddr_in servaddr;
if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
perror("socket creation failed");
exit(EXIT_FAILURE);
}
memset(&servaddr, 0, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(PORT);
servaddr.sin_addr.s_addr = INADDR_ANY;
int n, len;
sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM, (const struct sockaddr *) &servaddr,
sizeof(servaddr));
printf("Hello message sent.\n");
n = recvfrom(sockfd, (char *)buffer, MAXLINE,
MSG_WAITALL, (struct sockaddr *) &servaddr, &len);
buffer[n] = '\0';
printf("Server : %s\n", buffer);
close(sockfd);
return 0;
}
```

Server:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#define PORT 8080
#define MAXLINE 1024
int main() {
```

```
int sockfd;
char buffer[MAXLINE];
char *hello = "Hello from server";
struct sockaddr_in servaddr, cliaddr;
if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
perror("socket creation failed");
exit(EXIT_FAILURE);
}
memset(&servaddr, 0, sizeof(servaddr));
memset(&cliaddr, 0, sizeof(cliaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = INADDR_ANY;
servaddr.sin_port = htons(PORT);
if ( bind(sockfd, (const struct sockaddr *)&servaddr,
sizeof(servaddr)) < 0 )
{
perror("bind failed");
exit(EXIT_FAILURE);
}
int len, n;
len = sizeof(cliaddr);
n = recvfrom(sockfd, (char *)buffer, MAXLINE, MSG_WAITALL, ( struct sockaddr *) &cliaddr, &len);
buffer[n] = '\0';
printf("Client : %s\n", buffer);
sendto(sockfd, (const char *)hello, strlen(hello),
MSG_CONFIRM, (const struct sockaddr *) &cliaddr,len);
printf("Hello message sent.\n");
return 0;
}
```



```
~/ $ cd 5
~/5/ $ gcc client.c -o client
~/5/ $ ./client
Hello message sent.
Server : Hello from server
~/5/ $
```

```
~/ $ cd 5
~/5/ $ gcc server.c -o server
~/5/ $ ./server
Client : Hello from client
Hello message sent.
~/5/ $
```