UDP OUTPUTS:

client.c :

```
// Client side implementation of UDP client-server model
#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
// Driver code
int main() {
	int sockfd;
	char buffer[MAXLINE];
	char *hello = "Hello from client";
	struct sockaddr_in  servaddr;

	// Creating socket file descriptor
	if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
		perror("socket creation failed");
		exit(EXIT_FAILURE);
	}
	memset(&servaddr, 0, sizeof(servaddr));

	// Filling server information
	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.c :

```
// Server side implementation of UDP client-server model
#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
// Driver code
int main() {
        int sockfd;
        char buffer[MAXLINE];
        char *hello = "Hello from server";
        struct sockaddr_in servaddr, cliaddr;
        // Creating socket file descriptor
        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));
        // Filling server information
        servaddr.sin_family = AF_INET; // IPv4
        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); //len is value/result
        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;
}
```