CSE1004 NAC LAB
ASSIGNMENT NO. – 5

NETWORK CONFIGURATION COMMANDS

DATE – 03/01/22

NAME – AYUSHI TRIVEDI

REGISTRATION NUMBER – 20BPS1135

AIM: Send the following statement from the client and do the message encoding on the server side and display it on the client side.

"Please Switch Off the Lights and Fan When Not in Use" replace the vowels with their next vowel for example "a" with "e", "e" with "i" and so on. Similarly replace the consonants with the next consonant in order. Retain the Capital letters even after encoding.

SERVER:

ALGORITHM:

1. Create a UDP socket using socket() function
2. Then bind the socket to the server address using the bind() function
3. To receive the datagram from the client-side use receivefrom() function and store the message received in the buffer say "msg"
4. Use a for loop to iterate over the msg, then check the given char is a vowel or consonant using the user-defined function isVowelOrConsonant().
5. If this returns to 1, then its vowel and replace the given char with the next vowel using user-defined replaceVowel() function (Upper and Lowercase of char are considered in the function).
6. Else if it returns to 2, then it's consonant and replace the given char to the next consonant using user-defined replaceConsonant() function(Upper and Lowercase of char are considered in the function).
7. Then send the updated msg to the client-side using sendto() function.
8. Close the connection using close() function.

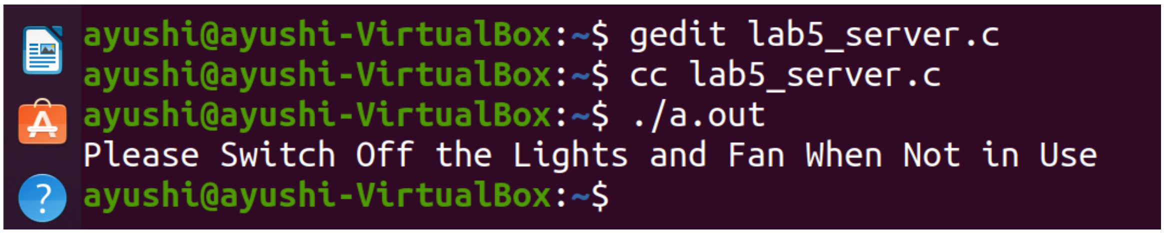
CODE:

```
// server program for udp connection
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <string.h>
#include <sys/socket.h>
#include<netinet/in.h>
#define PORT 5000
#define MAXLINE 1000
int isVowelOrConsonant(char ch){
if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' || ch ==
'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U'){
return 1;
}
else if (ch != ' '){
return 2;
}
}
char replaceVowel(char ch){
if (ch == 'a' || ch == 'e' || ch == 'A' || ch == 'E') return (ch + 4);
else if (ch == 'i' || ch == 'I') return (ch + 6);
else if (ch == 'o' || ch == 'O') return (ch + 6);
else if (ch == 'u') return 'a';
else if (ch == 'U') return 'A';
}
char replaceConsonant(char ch){
ch = ch + 1;
```

```
if ((int)ch == 91) ch = ch - 26;
if ((int)ch >= 122) ch = ch - 26;
if (isVowelOrConsonant(ch) == 1) return (ch + 1);
else return ch;
}
// Driver code
int main()
{
char buffer[500];
char *message = "Hello Client";
int listenfd, len;
struct sockaddr_in servaddr, cliaddr;
bzero(&servaddr, sizeof(servaddr));
// Create a UDP Socket
listenfd = socket(AF_INET, SOCK_DGRAM, 0);
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(PORT);
servaddr.sin_family = AF_INET;
// bind server address to socket descriptor
bind(listenfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
//receive the datagram
len = sizeof(cliaddr);
char msg[500];
int n = recvfrom(listenfd, msg, sizeof(msg),0, (struct
sockaddr*)&cliaddr,&len); //receive message from server
msg[n] = '\0';
puts(msg);
for(int i = 0; i < strlen(msg) ;i++){
if (isVowelOrConsonant(msg[i]) == 1){
```

```
msg[i] = replaceVowel(msg[i]);

}

else if (isVowelOrConsonant(msg[i]) == 2){

msg[i] = replaceConsonant(msg[i]);

}

}

// send the response

sendto(listenfd, msg, MAXLINE, 0,(struct sockaddr*)&cliaddr,

sizeof(cliaddr));

}
```

OUTPUT:



```
ayushi@ayushi-VirtualBox:~$ gedit lab5_server.c
ayushi@ayushi-VirtualBox:~$ cc lab5_server.c
ayushi@ayushi-VirtualBox:~$ ./a.out
Please Switch Off the Lights and Fan When Not in Use
ayushi@ayushi-VirtualBox:~$
```

CLIENT:

ALGORITHM:

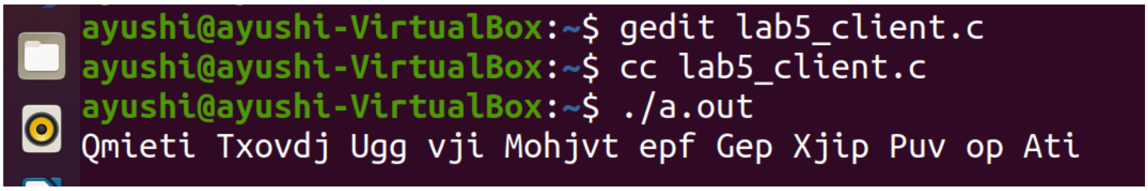
1. Create a UDP socket using socket() function
2. Then bind the socket to the server address using the bind() function.
3. Make a buffer say “msg” containing the sentence.
4. To send the datagram to the server-side use sendto() function.
5. Receive the updated msg from the server-side using receivefrom() function and store it in buffer (let’s say).
6. Print the received message.
7. Close the connection using close() function.

CODE:

```
// udp client driver program
#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include<netinet/in.h>
#include<unistd.h>
#include<stdlib.h>
#define PORT 5000
#define MAXLINE 1000
// Driver code
int main()
{
char buffer[500];
char *message = "Hello Server";
int sockfd, n;
struct sockaddr_in servaddr;
// clear servaddr
bzero(&servaddr, sizeof(servaddr));
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons(PORT);
servaddr.sin_family = AF_INET;
// create datagram socket
sockfd = socket(AF_INET, SOCK_DGRAM, 0);
// connect to server
if(connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
{
```

```
printf("\n Error : Connect Failed \n");
exit(0);
}
// request to send datagram
// no need to specify server address in sendto
// connect stores the peers IP and port
char msg[] = "Please Switch Off the Lights and Fan When Not in Use";
sendto(sockfd, msg, MAXLINE, 0, (struct sockaddr*)NULL, sizeof(servaddr));
// waiting for response
recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)NULL, NULL);
puts(buffer);
printf("\n");
// close the descriptor
close(sockfd);
}
```

OUTPUT:



```
ayushi@ayushi-VirtualBox:~$ gedit lab5_client.c
ayushi@ayushi-VirtualBox:~$ cc lab5_client.c
ayushi@ayushi-VirtualBox:~$ ./a.out
Qmieti Txovdj Ugg vji Mohjvt epf Gep Xjip Puv op Ati
```