**Assignment No : A 3 (b)**

**UDP**

**SOURCE CODE:**

UDP SERVER:

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <stdio.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <stdlib.h>

int main()

{

int sock;

int addr\_len, bytes\_read;

char recv\_data[1024];

struct sockaddr\_in server\_addr , client\_addr;

if ((sock = socket(AF\_INET, SOCK\_DGRAM, 0)) == -1)

{

perror("Socket");

exit(1);

}

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(5000);

server\_addr.sin\_addr.s\_addr = INADDR\_ANY;

bzero(&(server\_addr.sin\_zero),8);

if (bind(sock,(struct sockaddr \*)&server\_addr,

sizeof(struct sockaddr)) == -1)

{

perror("Bind");

exit(1);

}

addr\_len = sizeof(struct sockaddr);

printf("\nUDPServer Waiting for client on port 5000");

fflush(stdout);

while (1)

{

bytes\_read = recvfrom(sock,recv\_data,1024,0,

(struct sockaddr \*)&client\_addr, &addr\_len);

recv\_data[bytes\_read] = '\0';

printf("\n(%s , %d) said : ",inet\_ntoa(client\_addr.sin\_addr),ntohs(client\_addr.sin\_port));

printf("%s", recv\_data);

fflush(stdout);

}

return 0;

}

UDP CLIENT:

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <netdb.h>

#include <stdio.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <stdlib.h>

int main()

{

int sock;

struct sockaddr\_in server\_addr;

struct hostent \*host;

char send\_data[1024];

host= (struct hostent \*) gethostbyname((char \*)"127.0.0.1");

if ((sock = socket(AF\_INET, SOCK\_DGRAM, 0)) == -1)

{

perror("socket");

exit(1);

}

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(5000);

server\_addr.sin\_addr = \*((struct in\_addr \*)host->h\_addr);

bzero(&(server\_addr.sin\_zero),8);

while (1)

{

printf("Type Something (q or Q to quit):");

gets(send\_data);

if ((strcmp(send\_data , "q") == 0) || strcmp(send\_data , "Q") == 0)

break;

else

sendto(sock, send\_data, strlen(send\_data), 0,

(struct sockaddr \*)&server\_addr, sizeof(struct sockaddr));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



