Ex.No:5	CONCURRENT TCP/IP DAY-TIME SERVER
Date:	

Aim:

There are two hosts, Client and Server. The Client requests the concurrent server for the date and time. The Server sends the date and time, which the Client accepts and prints.

TECHNICAL OBJECTIVE:

To implement a TCP/IP day time server (concurrent server) that handles multiple client requests. Once the client establishes connection with the server, the server sends its day-time details to the client which the client prints in its console.

METHODOLOGY:

TCP Server :

- Create Socket address structure.
- TCP/UDP socket is created, an Internet socket address structure is filled with wildcard address & server's well known port.
- Bind function assigns a local protocol address to the socket.
- Listen function specifies the maximum number of connections that kernel should queue for this socket.
- The server to return the next completed connection from the front of the completed connection Queue calls it.
- Receiving the request message from the client.

TCP Client :

- This function looks up a hostname and it returns a pointer to a hostent structure that contains all the IPV4 address.
- Create Socket address structure. Creating a socket, assigning IP address and port number for that socket.
- Connect establishes connection with the server using server IP address.
- Reads the message from standard input.
- Send function is used on client side to send data given by user on client side to the server.

Server:

```
#include<netinet/in.h>
#include<sys/socket.h>
#include<stdio.h>
#include<string.h>
#include<time.h>
#include<stdlib.h>
int main()
{
        struct sockaddr_in sa;
        struct sockaddr_in cli;
```

```
		int sockfd,conntfd;
		int len,ch;
		char str[100];
		time_t tick;
		sockfd=socket(AF_INET,SOCK_STREAM,0);
	if(sockfd<0)
	{
		printf("error in socket\n");
		exit(0);
	}
	else
		printf("Socket opened");
		bzero(&sa,sizeof(sa));
		sa.sin_port=htons(5600);
		sa.sin_addr.s_addr=htonl(0);
	if(bind(sockfd,(struct sockaddr*)&sa,sizeof(sa))<0)
	{
		printf("Error in binding\n");
	}
	else
		printf("Binded Successfully");
		listen(sockfd,50);
	for(;;)
	{
		len=sizeof(ch);
		conntfd=accept(sockfd,(struct sockaddr*)&cli,&len);
		printf("Accepted");
		tick=time(NULL);
		snprintf(str,sizeof(str),"%s",ctime(&tick));
		printf("%s",str);write(conntfd,str,100);
	}
	}
```

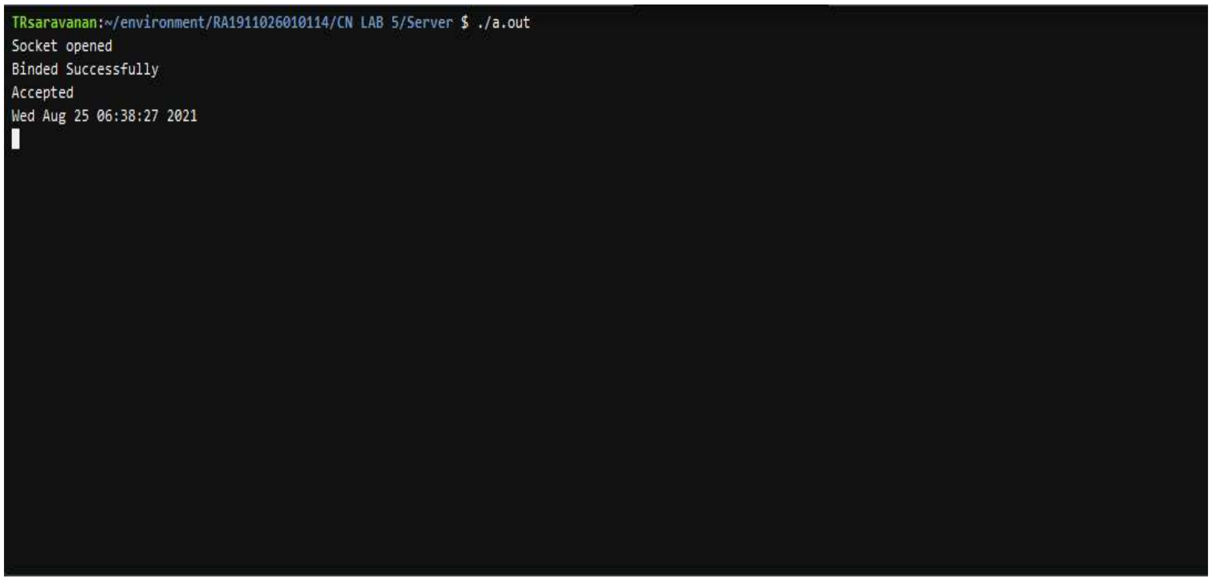
Client:

```
	#include <netinet/in.h>
	#include <sys/socket.h>
	#include <stdio.h>
	#include <stdlib.h>
	int main()
	{
		struct sockaddr_in sa,cli;
		int n,sockfd;
		int len;char buff[100];
		sockfd=socket(AF_INET,SOCK_STREAM,0);
		if(sockfd<0)
	{
```

```
                 printf("\nError in Socket");
                exit(0);
        }
        else
                printf("\nSocket is Opened");
                bzero(&sa,sizeof(sa));
                sa.sin_family=AF_INET;
                sa.sin_port=htons(5600);
        if(connect(sockfd,(struct sockaddr*)&sa,sizeof(sa))<0)
        {
                printf("\nError in connection failed");
                exit(0);
        }
        else
                printf("\nconnected successfully");
        if(n=read(sockfd,buff,sizeof(buff))<0)
        {
                printf("\nError in Reading");
                exit(0);
        }
        else
        {
                printf("\nMessage Read %s",buff);
        }
        }
```

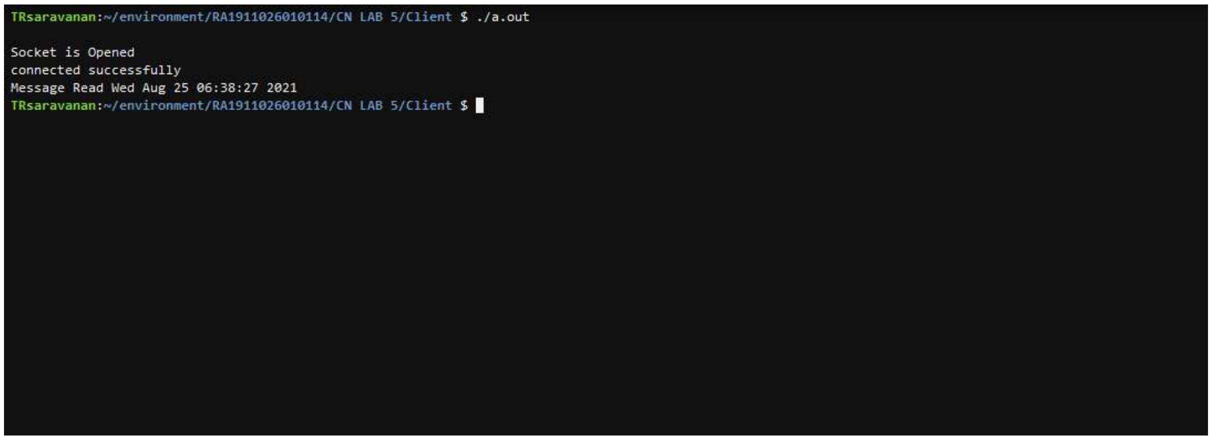
Sample Output:

Server Side :



```
TRsaravanan:~/environment/RA1911026010114/CN LAB 5/Server $ ./a.out
Socket opened
Binded Successfully
Accepted
Wed Aug 25 06:38:27 2021
█
```

Client Side :



```
TRsaravanan:~/environment/RA1911026010114/CN LAB 5/Client $ ./a.out

Socket is Opened
connected successfully
Message Read Wed Aug 25 06:38:27 2021
TRsaravanan:~/environment/RA1911026010114/CN LAB 5/Client $ 
```

Result :

Thus the concurrent daytime client- server communication is established by sending the request message from the client to the concurrent server and the server sends its time to all the clients and displays it.