Computer Networks Lab- Experiment 2

Name: Rahul Goel
Reg No: RA1911030010094
Section- O2

SERVER CODE :

```
// Server side C/C++ program to demonstrate Socket programming
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
int main(int argc, char const *argv[])
{
	int server_fd, new_socket, valread;
	struct sockaddr_in address;
	int opt = 1;
	int addrlen = sizeof(address);
	char buffer[1024] = {0};
	char *hello = "Hello from server";

	// Creating socket file descriptor
	if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
	{
		perror("socket failed");
		exit(EXIT_FAILURE);
	}

	// Forcefully attaching socket to the port 8080
	if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,

&opt, sizeof(opt)))
	{
		perror("setsockopt");
		exit(EXIT_FAILURE);
	}
	address.sin_family = AF_INET;
	address.sin_addr.s_addr = INADDR_ANY;
	address.sin_port = htons( PORT );

	// Forcefully attaching socket to the port 8080
	if (bind(server_fd, (struct sockaddr *)&address,
							sizeof(address))<0)
```

```
        {
                perror("bind failed");
                exit(EXIT_FAILURE);
        }
        if (listen(server_fd, 3) < 0)
        {
                perror("listen");
                exit(EXIT_FAILURE);
        }
        if ((new_socket = accept(server_fd, (struct sockaddr *)&address,
                                        (socklen_t*)&addrlen))<0)
        {
                perror("accept");
                exit(EXIT_FAILURE);
        }
        valread = read( new_socket , buffer, 1024);
        printf("%s\n",buffer );
        send(new_socket , hello , strlen(hello) , 0 );
        printf("Hello message sent\n");
        return 0;
}
```

```
// Client side C/C++ program to demonstrate Socket programming
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080

int main(int argc, char const *argv[])
{
        int sock = 0, valread;
        struct sockaddr_in serv_addr;
        char *hello = "Hello from client";
        char buffer[1024] = {0};
        if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
        {
                printf("\n Socket creation error \n");
                return -1;
        }

        serv_addr.sin_family = AF_INET;
        serv_addr.sin_port = htons(PORT);

        // Convert IPv4 and IPv6 addresses from text to binary form
        if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
        {
                printf("\nInvalid address/ Address not supported \n");
                return -1;
        }

        if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
        {
                printf("\nConnection Failed \n");
                return -1;
        }
        send(sock , hello , strlen(hello) , 0 );
        printf("Hello message sent\n");
        valread = read( sock , buffer, 1024);
        printf("%s\n",buffer );
        return 0;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"client.c" 42L, 1009B
```

CLIENT
CODE :

```
// Client side C/C++ program to demonstrate Socket programming
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080

int main(int argc, char const *argv[])
{
	int sock = 0, valread;
	struct sockaddr_in serv_addr;
	char *hello = "Hello from client";
	char buffer[1024] = {0};
	if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
	{
		printf("\n Socket creation error \n");
		return -1;
	}

	serv_addr.sin_family = AF_INET;
	serv_addr.sin_port = htons(PORT);

	// Convert IPv4 and IPv6 addresses from text to binary form
	if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
	{
		printf("\nInvalid address/ Address not supported \n");
		return -1;
	}

	if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
	{
		printf("\nConnection Failed \n");
		return -1;
	}
	send(sock , hello , strlen(hello) , 0 );
	printf("Hello message sent\n");
	valread = read( sock , buffer, 1024);
	printf("%s\n",buffer );
	return 0;
}
```

```c
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
int main(int argc, char const *argv[])
{
        int server_fd, new_socket, valread;
        struct sockaddr_in address;
        int opt = 1;
        int addrlen = sizeof(address);
        char buffer[1024] = {0};
        char *hello = "Hello from server";

        // Creating socket file descriptor
        if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
        {
                perror("socket failed");
                exit(EXIT_FAILURE);
        }

        // Forcefully attaching socket to the port 8080
        if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,
                                                                                &opt, sizeof(opt)))
        {
                perror("setsockopt");
                exit(EXIT_FAILURE);
        }
        address.sin_family = AF_INET;
        address.sin_addr.s_addr = INADDR_ANY;
        address.sin_port = htons( PORT );

        // Forcefully attaching socket to the port 8080
        if (bind(server_fd, (struct sockaddr *)&address,
                                                        sizeof(address))<0)
        {
                perror("bind failed");
                exit(EXIT_FAILURE);
        }
        if (listen(server_fd, 3) < 0)
        {
                perror("listen");
                exit(EXIT_FAILURE);
        }
        if ((new_socket = accept(server_fd, (struct sockaddr *)&address,
                                        (socklen_t*)&addrlen))<0)
        {
                perror("accept");
                exit(EXIT_FAILURE);
        }
        valread = read( new_socket , buffer, 1024);
        printf("%s\n",buffer );
        send(new_socket , hello , strlen(hello) , 0 );
        printf("Hello message sent\n");
        return 0;
}
```