EXCERSISE 1

SOCKET PROGRAMMING

SERVER.C

```c
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>

#define MAXSIZE 1024

int main(int argc, char *argv[])
{
    if (argc < 3)
    {
        printf("Enter the port number and the string to send\n");
        return 1;
    }
    int server_fd, new_socket, valread;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);

    char buffer[MAXSIZE] = {0};
    int PORT = atoi(argv[1]);
    char *msg = argv[2];

    // 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 given port
    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);

    // binding socket to the port
    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, MAXSIZE);
    printf("Message from client: %s\n", buffer);
    send(new_socket, msg, strlen(msg), 0);
    printf("Message has been sent to the client\n");

    return 0;
}
```

CLIENT.C

```c
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

#define MAXSIZE 1024

int main(int argc, char *argv[])
{
    if (argc < 3)
    {
        printf("Enter the port number and the string to send\n");
        return 1;
    }

    int sock = 0, valread;
    struct sockaddr_in serv_addr;

    int PORT = atoi(argv[1]);
    char *msg = argv[2];

    char buffer[MAXSIZE] = {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);

    // Converting 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, msg, strlen(msg), 0);
    printf("Your message has been sent to server\n");

    valread = read(sock, buffer, MAXSIZE);
    printf("Welcome %s\n", buffer);

    return 0;
}
```

OUTPUT :