* Implement Simple Calculator using client-server Communication. Write a program to create UDP socket.

→ server.c

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

#define PORT 8080
#define BUFFER_SIZE 1024

double perform_operation(char operation, double num1,
                         double num2)
{
  switch (operation)
  {
    case '+':
        return num1 + num2;
    case '-':
        return num1 - num2;
    case '*':
        return num1 * num2;
    case '/':
        if (num2 != 0) return num1 / num2;
        else return 0;
    default:
        return 0.0;
  }
}
```

```
int main()
{
  int sockfd;
  struct sockaddr_in server_addr, client_addr;
  char buffer[BUFFER_SIZE];
  char operation;
  double num1, num2, result;
  socklen_t addr_len = sizeof(client_addr);

  if((sockfd = socket(AF_INET, SOCK_DGRAM, 0))<0)
  {
    perror("Socket creation failed");
    exit(EXIT_FAILURE);
  }

  memset(&addserver_addr, 0, sizeof(server_addr));
  memset(&client_addr, 0, sizeof(client_addr));

  server_addr.sin_family = AF_INET;
  server_addr.sin_addr.s_addr = INADDR_ANY;
  server_addr.sin_port = htons(PORT);

  if(bind(sockfd, (const struct sockaddr *)&server_addr,
                    sizeof(server_addr)) < 0)
  {
    perror("Bind failed");
    close(sockfd);
    exit(EXIT_FAILURE);
  }
  printf("Server is listening on port %d\n", PORT);
```

```
recvfrom(sockfd, (char *)buffer, BUFFER_SIZE,
        MSG_WAITALL, (struct sockaddr *) &client_addr,
                                &addr_len);
sscanf(buffer, "%lf %c %lf", &num1, &operation,
                                &num2);

result = perform_opperation(operation, num1, num2);

snprintf(buffer, BUFFER_SIZE, "Result: %.2lf", result);
sendto(sockfd, (const char *)buffer, strlen(buffer),
    MSG_CONFIRM, (const struct sockaddr *)
        &client_addr, addr_len);

close(sockfd);

return 0;
}
```

client.c

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

#define PORT 8080
#define BUFFER_SIZE 1024

int main()
{
  int sockfd;
  struct sockaddr_in serv_addr;
  char buffer[BUFFER_SIZE];
  char operation;
  double num1, num2;
  socklen_t addr_len;

  if((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
  {
    printf("Socket creation error\n");
    return -1;
  }

  memset(&serv_addr, 0, sizeof(serv_addr));

  serv_addr.sin_family = AF_INET;
  serv_addr.sin_port = htons(PORT);
  serv_addr.sin_addr.s_addr = INADDR_ANY;
```

```
printf ("Enter operation : ");
scanf ("%lf %c %lf", &num1, &operation, &num2);

snprintf (buffer, BUFFER_SIZE, "%lf %c %lf", num1,
                                operation, num2);
sendto (sockfd, (const char *) buffer, strlen (buffer),
    MSG_CONFIRM, (const struct sockaddr *) &serv_
    addr, sizeof (serv_addr));

addr_len = sizeof (serv_addr);
recvfrom ( sockfd, (char *) buffer, BUFFER_SIZE,
 MSG_WAITALL, (struct sockaddr *) &serv_addr,
                    &addr_len );

printf ("%s \n", buffer);

close (sockfd);

return 0;
}
```