CN ASSIGNMENT

Name – Shruti Sood

PRN no – 12210002

Div – IT-C

Roll no. 42

Code-

Client side

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8080

int main() {

int client\_socket;

struct sockaddr\_in server\_address;

if ((client\_socket = socket(AF\_INET, SOCK\_STREAM, 0)) < 0) {

perror("socket failed");

exit(EXIT\_FAILURE);

}

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

server\_address.sin\_port = htons(PORT);

if (inet\_pton(AF\_INET, "127.0.0.1", &server\_address.sin\_addr) <= 0) {

perror("Invalid address / Address not supported");

exit(EXIT\_FAILURE);

}

if (connect(client\_socket, (struct sockaddr \*)&server\_address, sizeof(server\_address)) < 0) {

perror("connection failed");

exit(EXIT\_FAILURE);

}

int num1, num2, result;

char operator;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter an operator (+, -, \*, /): ");

scanf(" %c", &operator);

printf("Enter second number: ");

scanf("%d", &num2);

write(client\_socket, &num1, sizeof(int));

write(client\_socket, &num2, sizeof(int));

write(client\_socket, &operator, sizeof(char));

read(client\_socket, &result, sizeof(int));

if (result == -1) {

printf("Error: Invalid operator or division by zero.\n");

} else {

printf("Result: %d %c %d = %d\n", num1, operator, num2, result);

}

close(client\_socket);

return 0;

}

Server side

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8080

int calculate(int a, int b, char operator) {

switch (operator) {

case '+':

return a + b;

case '-':

return a - b;

case '\*':

return a \* b;

case '/':

return b != 0 ? a / b : -1; // Avoid division by zero

default:

return -1; // Invalid operator

}

}

int main() {

int server\_fd, new\_socket;

struct sockaddr\_in address;

int addrlen = sizeof(address);

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

perror("socket failed");

exit(EXIT\_FAILURE);

}

address.sin\_family = AF\_INET;

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

address.sin\_port = htons(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);

}

printf("Server is listening on port %d\n", PORT);

while (1) {

if ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address, (socklen\_t \*)&addrlen)) < 0) {

perror("accept");

exit(EXIT\_FAILURE);

}

int num1, num2, result;

char operator;

read(new\_socket, &num1, sizeof(int));

read(new\_socket, &num2, sizeof(int));

read(new\_socket, &operator, sizeof(char));

result = calculate(num1, num2, operator);

printf("Received: %d %c %d = %d\n", num1, operator, num2, result);

send(new\_socket, &result, sizeof(int), 0);

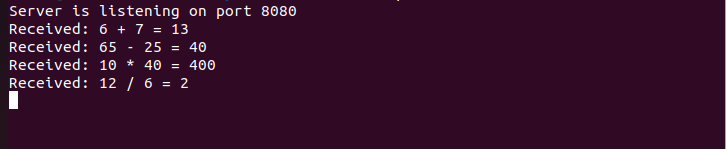
close(new\_socket);

}

return 0;

}

Output





A black background with white text

Description automatically generated