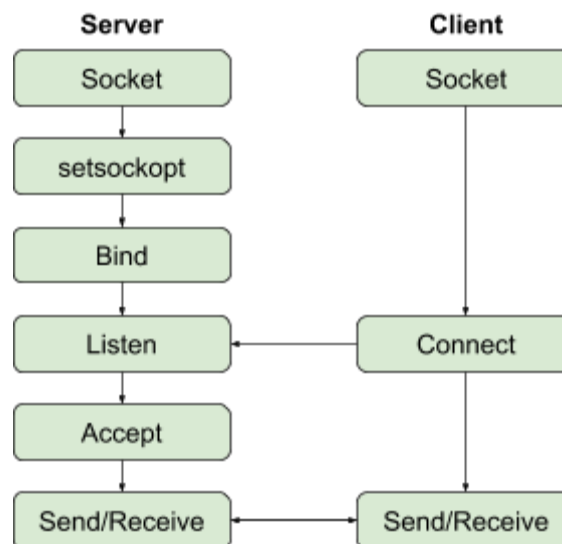


Assignment 2

1st August, 2019

1. Write a socket program in C to connect two nodes on a network to communicate with each other, where one socket listens on a particular port at an IP, while other socket reaches out to the other to form a connection.



Server Side:

```
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 *greetings = "Greetings from server";

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

```

}

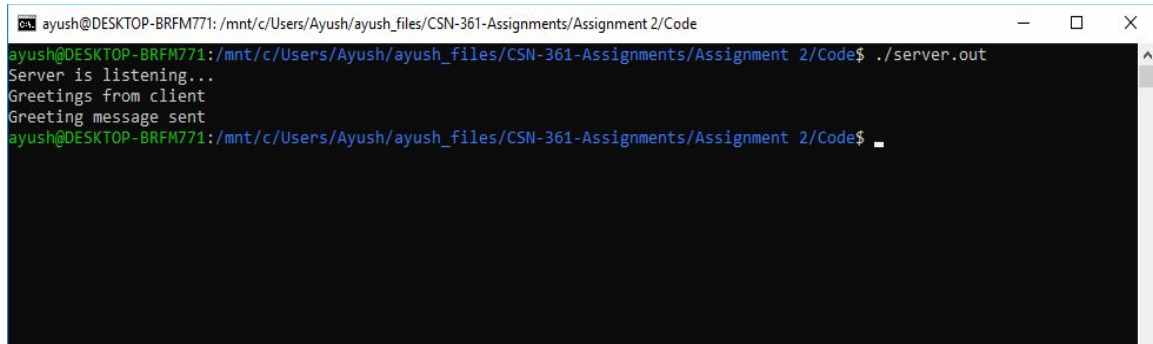
// Forcefully attaching socket to the port 8080
if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,
               &opt, sizeof(opt)))
{
    perror("setsockopt");
    exit(EXIT_FAILURE);
}
address.sin_family = AF_INET;
address.sin_addr.s_addr = INADDR_ANY;
address.sin_port = htons(PORT);

// Forcefully attaching socket to the port 8080
if (bind(server_fd, (struct sockaddr *)&address,
         sizeof(address)) < 0)
{
    perror("bind failed");
    exit(EXIT_FAILURE);
}

printf("Server is listening...\n");

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, greetings, strlen(greetings), 0);
printf("Greeting message sent\n");
return 0;
}

```



A terminal window titled "ayush@DESKTOP-BRFM771: /mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code" showing the execution of the server program. The prompt is "ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code\$". The user enters "./server.out". The output shows "Server is listening...", "Greetings from client", and "Greeting message sent". The prompt returns to "ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code\$".

```

ayush@DESKTOP-BRFM771: /mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$ ./server.out
Server is listening...
Greetings from client
Greeting message sent
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$

```

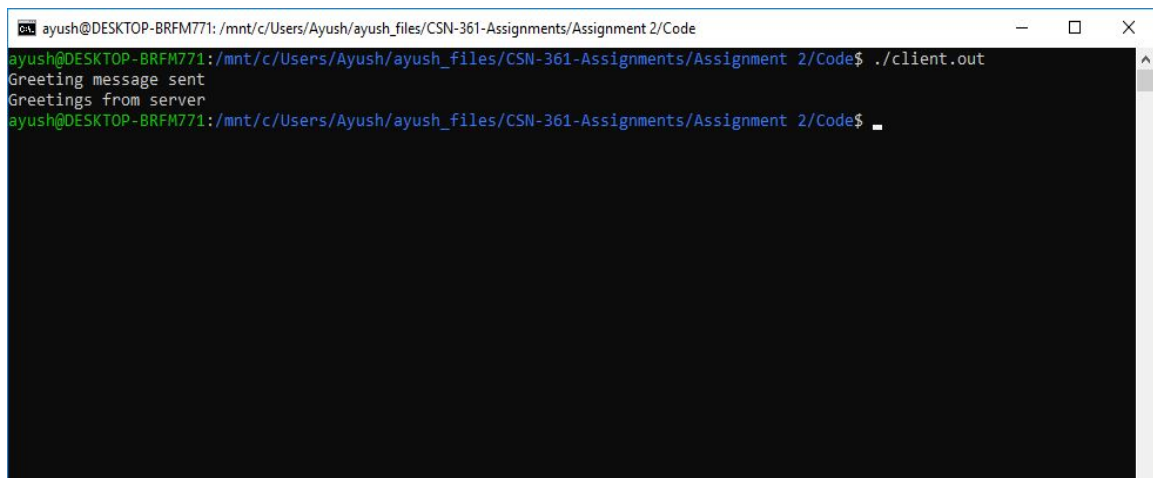
Client Side:

```
int main(int argc, char const *argv[])
{
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char *greetings = "Greetings 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, greetings, strlen(greetings), 0);
    printf("Greeting message sent\n");
    valread = read(sock, buffer, 1024);
    printf("%s\n", buffer);
    return 0;
}
```



A terminal window titled "ayush@DESKTOP-BRFM771: /mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code" is shown. The prompt is "ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code\$". The user has entered the command "./client.out". The output of the program is displayed on two lines: "Greeting message sent" and "Greetings from server". The prompt is now "ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code\$ " with a cursor.

-
2. Write a C++ program to demonstrate both Zombie and the Orphan process.

Zombie Process: A process that has finished the execution but still has an entry in the process table to report to its parent process is known as a zombie process.

Orphan Process: A process whose parent process no more exists i.e. either finished or terminated without waiting for its child process to terminate is called an orphan process.

```
int main()
{
    cout << "Parent pid: " << getpid() << endl << endl;

    pid_t child_pid = fork(); // created child from parent

    if (child_pid > 0)
    { // parent is executing
        cout << "Parent is active..." << endl;
        sleep(4);
        cout << "Parent is terminated after 2 seconds" << endl;
    }

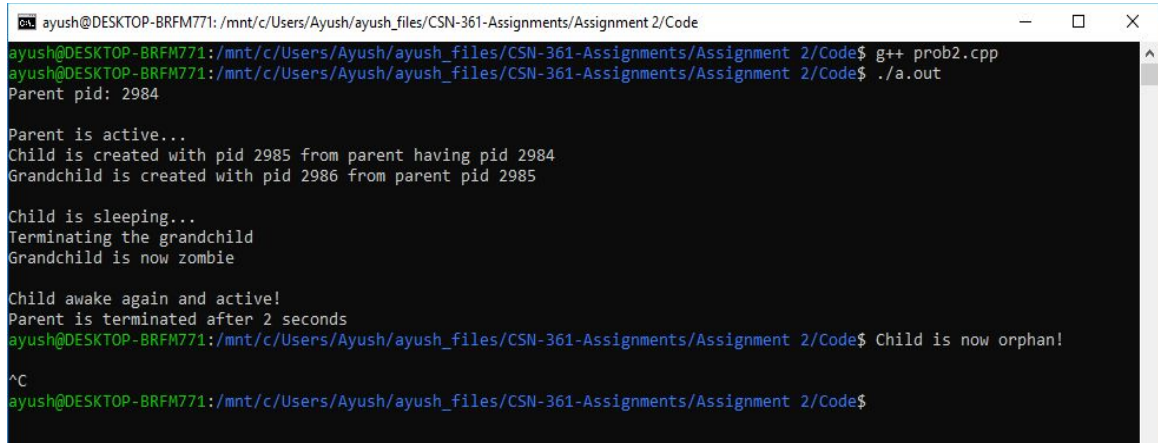
    else if (child_pid == 0) // child is executing
    {
        cout << "Child is created with pid " << getpid()
        << " from parent having pid " << getppid() << endl;

        child_pid = fork(); // grandchild is created

        if (child_pid > 0) // child is executing
        {
            sleep(1);
            cout << "Child is sleeping..." << endl;
            sleep(2);
            cout << "Child awake again and active!" << endl;
            sleep(2);
            cout << "Child is now orphan!" << endl
            << endl;
        }

        else if (child_pid == 0)
        {
            cout << "Grandchild is created with pid " << getpid()
            << " from parent pid " << getppid() << endl << endl;
            sleep(1);
            cout << "Terminating the grandchild" << endl;
        }
    }
}
```

```
        cout << "Grandchild is now zombie" << endl << endl;  
    }  
}  
  
return 0;  
}
```



```
ayush@DESKTOP-BRFM771: /mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code  
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$ g++ prob2.cpp  
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$ ./a.out  
Parent pid: 2984  
  
Parent is active...  
Child is created with pid 2985 from parent having pid 2984  
Grandchild is created with pid 2986 from parent pid 2985  
  
Child is sleeping...  
Terminating the grandchild  
Grandchild is now zombie  
  
Child awake again and active!  
Parent is terminated after 2 seconds  
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$ Child is now orphan!  
^C  
ayush@DESKTOP-BRFM771:/mnt/c/Users/Ayush/ayush_files/CSN-361-Assignments/Assignment 2/Code$
```