Date: 24/08/2024

Lab Practical #07:

Study Client-Server Socket programming - TCP & UDP

Practical Assignment #07:

1. **Write a C/Java code for TCP Server-Client Socket Programming.**
2. **Write a C/Java code for UDP Server-Client Socket Programming.**

1. For TCP Server-Client:

TCP Server Program:

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

#define PORT 8080

int main() {
  int server_fd, new_socket;
  struct sockaddr_in address;
  int addrlen = sizeof(address);
  char buffer[1024] = {0};
  char *hello = "Hello from server";

  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);
```

Date: 24/08/2024

```
    }

    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);
    }

    read(new_socket, buffer, 1024);
    printf("Message from client: %s\n", buffer);
    send(new_socket, hello, strlen(hello), 0);
    printf("Hello message sent\n");
    close(new_socket);
    close(server_fd);
    return 0;
}
```

TCP Client Program:

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

#define PORT 8080

int main() {
    int sock = 0;
    struct sockaddr_in serv_addr;
    char *hello = "Hello from client";
    char buffer[1024] = {0};

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
```

Date: 24/08/2024

```
        printf("\n Socket creation error \n");
        return -1;
    }

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

    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");
    read(sock, buffer, 1024);
    printf("Message from server: %s\n", buffer);
    close(sock);
    return 0;
}
```

Date: 24/08/2024

2. For UDP Server-Client:

UDP Server Program:

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

#define PORT 8080

int main() {
  int sockfd;
  char buffer[1024];
  char *hello = "Hello from server";
  struct sockaddr_in servaddr, cliaddr;

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

  memset(&servaddr, 0, sizeof(servaddr));
  memset(&cliaddr, 0, sizeof(cliaddr));

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

  if (bind(sockfd, (const struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
    perror("Bind failed");
    exit(EXIT_FAILURE);
  }

  int len, n;
  len = sizeof(cliaddr);
```

Date: 24/08/2024

```
    n = recvfrom(sockfd, (char *)buffer, 1024, MSG_WAITALL, (struct sockaddr *)&cliaddr, &len);
    buffer[n] = '\0';
    printf("Message from client: %s\n", buffer);
    sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM, (const struct sockaddr *)&cliaddr, len);
    printf("Hello message sent\n");

    close(sockfd);
    return 0;
}
```

UDP Client Program:

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

#define PORT 8080

int main() {
    int sockfd;
    char buffer[1024];
    char *hello = "Hello from client";
    struct sockaddr_in servaddr;

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

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

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

Date: 24/08/2024

```
    int n, len;

    sendto(sockfd, (const char *)hello, strlen(hello), MSG_CONFIRM, (const struct sockaddr *)&servaddr,
sizeof(servaddr));
    printf("Hello message sent\n");

    n = recvfrom(sockfd, (char *)buffer, 1024, MSG_WAITALL, (struct sockaddr *)&servaddr, &len);
    buffer[n] = '\0';
    printf("Message from server: %s\n", buffer);

    close(sockfd);
    return 0;
}
```