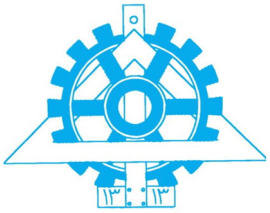
به نام خدا

تمرین کامپیوتری ۱

دانشکده مهندسی برق و کامپیوتر دانشگاه تهران

شبکه های کامپیوتری، نیم‌سال دوم سال تحصیلی ۱۰۰-۱



در این پروژه، هدف آشنایی با مفاهیم اولیه Socket Programming و نحوه ارتباط بین چندین برنامه است است که همه بر روی یک Local Host ولی با Portهای مختلف هستند.

در این قسمت سرور که یک Chatroom ساده است، به همراه نحوه ارتباط با آن در اختیار شما قرار می گیرد و شما یک کلاینت برای آن پیاده سازی می کنید. هر کلاینت یک ارتباط TCP با سرور برقرار می کند و این ارتباط باید تا انتهای برنامه باز باشد.

ساختار کلی پیام‌ها و پاسخ‌های سرور به صورت زیر است:

Message Type (4b)	Message ID (4b)	Length (1B)
Payload		

در این ساختار Message Typeمی‌توان مقادیر جدول ۱ را داشته باشد که درقسمت‌های بعدی هر کدام معرفی شده‌اند. Message ID یک عدد متفاوت برای هر پیام است که امکان می‌دهد کانکشن TCP بین پیام‌های مختلف به اشتراک گذاشته شود. اگر پس از هر درخواست منتظر پاسخ سرور می‌مانید مقدار Message ID اهمیتی ندارد. Length هم طول کل پیام را مشخص می‌کند.

Message Type	Value
CONNECT	1
CONNACK	2
LIST	3
LISTREPLY	4
INFO	5
INFOREPLY	6
SEND	7
SENDREPLY	8
RECEIVE	9

RECEIVEREPLY	10

- کلاینت پس از اتصال به سرور اسم خود را با استفاده از پیام CONNECT برای سرور ارسال می‌کند. در پاسخ سرور پیام CONNACK را ارسال می‌کند. در صورت عدم ارسال این دستور کانکشن توسط سرور بسته می شود. تا زمان باز بودن ارتباط کلاینت این نام در سرور معتبر باقی خواهد ماند.

CONNECT	Message ID	2+length(name)
name		

CONNACK	Message ID	2

- هر کلاینت می تواند با پیام LIST شناسه تمام کاربران را از سرور درخواست کند. در پاخ به این دستور سرور لیست تمام کاربران را برای کلاینت با پیام LISTREPLY ارسال می‌کند. هر عضو این لیست شامل یک عدد 2بایتی به عنوان شناسه کاربر است.

LIST	Message ID	2

LISTREPLAY	Message ID	2+2*n
User ID (2B) * n		

- هر کلاینت می تواند با پیام INFO شناسه کاربر را به سرور ارسال کند و نام کاربر را بدست آورد. در پاسخ سرور پیام INFOREPLY را برای کلاینت ارسال می‌کند و در Payload آن، نام کاربر قرار دارد. اگر طول Payload صفر باشد یعنی کاربر در سرور وجود ندارد.

USERINFO	Message ID	2 + 2
User ID		

USERINFOREPLY	Message ID	2+length(user name)

User Name

- کلاینت برای ارسال پیام از دستور SEND استفاده می‌کند. این پیام شامل شناسه کاربری که پیام باید برایش ارسال شود و متن پیام است. سرور با پیام SENDREPLY پاسخ می‌دهد. وضعیت ارسال پیام در پاسخ وجود دارد.

SEND	Message ID	2 + 2+length(message)
User ID		
Message		

SENDREPLY	Message ID	2+1
Status (0: success, 1: failure)		

- کلاینت برای دریافت پیام‌هایی که برایش ارسال شده اند، از پیام RECEIVE استفاده می‌کند. سرور در پاسخ از پیام RECEIVEREPLY استفاده می‌کند و پیامی که برای کاربر ارسال شده است را برای کلاینت می‌فرستد. اگر پیامی موجود نباشد،Sender ID برابر 0 و Payload خالی خواهد بود.

RECEIVE	Message ID	2

RECEIVEREPLY	Message ID	2+2+length(message)
Sender ID		
Message		

انتظارات از کلاینت:

برنامه کلاینت در زمان اجرا آدرس سرور (با فرمت HOST:PORT) و نام کاربر را به عنوان آرگمان می گیرد.
برنامه کلاینت باید سه دستور را در اختیار کاربر قرار دهد.

- List: کلاینت از سرور لیست همه کاربران را دریافت می کند و نمایش می دهد.

>> list
- <User Name 1>
- <User Name 2>
...

- Send: پیغامی برای یکی از کلاینت ها ارسال کند. این دستور دو ورودی دارد که یکی نام گیرنده است و دیگری پیغام.

>> send <User Name> <Message>

- Exit: از برنامه خارج می شود.

در هر زمان که پیامی از دیگران برای کلاینت ارسال شد برنامه باید آن را به شکل زیر نمایش دهد.

<< <User Name>: <Message>

در ادامه مثالی از اجرای برنامه آورده شده است:

```
$ ./client localhost:9000 ali
>> list
    - sirous
>> send sirous Hello
<< sirous: Hi
>> exit
```