به نام خدا

پروژه درس شبکه های کامپیوتری – بخش اول

محمد سعید زارع – محمد مهدی فلاح

در این بخش قصد داریم یک ماشین حساب توزیع شده بنویسیم که محاسبات را روی سرور انجام دهد و سپس نتیجه را به کلاینت برگرداند. این پروژه از سه فایل Client.java و Server.java و ClientHandler.java در ادامه به بررسی ساختار فایل ها و همچنین کارکرد برخی توابع مهم خواهیم پرداخت.

فایل Client.java: این فایل مربوط به عملکرد کلاینت است. کلاس Client حاوی ۳ ویژگی socket، bufferedReader و bufferedWrite است که socket برای برقراری یک ارتباط امن (TCP) با سرور است. و دیگر موارد مربوط به نوشتن و خواندن فایل از stream مربوط به ارتباط کلاینت با سرور است. (به این دلیل از TCP Connection استفاده کردیم که بیشتر از سرعت در دریافت و ارسال پیام به یک ارتباط reliable احتیاج داشتیم که اطمینان داشته باشیم پیام ها توسط سرور دریافت و نتایج درستی به ما برگردانده شود. همچنین میدانیم که ماشین حساب از برنامه های live streaming نیست.)

برای ساخت یک ارتباط TCP در Client مقدار Hostname و port سرور دریافت میشود و سپس یک socket برای ارتباط با سرور ایجاد میشود. دو تابع کمکی داریم یکی sendMessage و دیگری closeEveryThing . تابع اولی برای ارسال پیام به سمت سرور است و پیام را روی stream مینویسد و سپس flush میکند و تابع دوم همه resourse های گرفته شده توسط کلاینت مثل socket، reader و writer را میبندد. از توابع اصلی میتوان به listenForResponse و sendCommanToServer اشاره کرد. تابع اول بصورت چند نخی پیام ها را از سرور دریافت میکند و نشان میدهد و همچنین تابع دوم کامند نوشته شده در کنسول را به سرور ارسال میکند.

فایل Server.java: این کلاس با دریافت پورت serverSocket یک سوکت سروری ایجاد میکند. همچنین تابع اصلی این کلاس startServer است که در یک حلقه بی نهایت کلاینت هایی که به سرور درخواست میدهند را accept میکند و سپس یک نمونه از کلاس clientHandler ایجاد میکند و سوکت مربوط به کلاینت را به آن پاس میدهد. پس از اینجای کار clientHandler مسئول رسیدگی به درخواست های client خواهد بود و server پس ایجاد این thread باز منتظر دریافت درخواست از طوف کلاینت ها خواهد بود. پس با توجه به این که اینجا سرور تا انجام درخواست کلاینت بلاک نخواهد شد و قابلیت دریافت چند کلاینت بصورت همزمان را دارد پس این برنامه multi-thread است. این فایل حاوی دو تابع closeServerSocket که جهت بستن سوکت سرور است و دیگری broadCast که نوشته شده تا اگر سرور بخواهد به همه یک پیامی مخابره کنده بتواند.

فایل ClientHandler.java: این کلاس از Runnable ارث بری میکند زیرا نمونه های این کلاس باید بصورت چند نخی اجرا شوند. این کلاس علاوه بر ویژگی های کلاس Client یک لیستی از clientHandler ها را در خودش نگه میدارد و این به دلیل این است که اگر بخواهیم broadCastی انجام دهیم از این لیست استفاده کنیم. این کلاس حاوی یکسری متد نام آشنا مانند closeEveryThing، sendMessage و removeClientHandler است که کلاینت جاری را از لیست clientHandler ها خارج میکند.

منطق اصلی کارکرد این کلاس در تابع run نوشته شده است که ابتدا کامند ارسالی توسط کلاینت گرفته میشود و سپس پارس میشود. (ابتدا split میشود و سپس هرکدام از بخش ها بررسی میشوند که مطمئن شویم کامند ارسال شده به فرمت درستی است. برای مثال مقدار operator های تک عملونده و دوعملونده بصورت مجزا در لیست هایی ذخیره شده اند و operator ارسالی با اینها چک میشود همچنین یک RegEx نوشته شده تا چیز غیر تبدیل به عدد را شناسایی کند و برای کاربر error برگرداند. برای مثال اگر بجای عدد string وارد شود. سرور به کاربر خطا خواهد داد. فرمت خطای ارسال به صورت "Err 0" است.) سپس بعد از پارس، timer به نانوثانیه زمان فعلی را اندازه میگیرد و عملیات انجام خواهد شد و سپس بعد از انجام عملیات باز هم زمان تایمر بررسی میشود و خلاصه اختلاف این دو زمان بررسی میشود. و در نتیجه، نتیجه محاسبه در صورت نبود خطا و همچنین زمان گرفته شده برای انجام محاسبه به کلاینت ارسال خواهد شد. **(زاویه ارسالی باید به صورت درجه باشد.)**

نکته: هم خطاهای مربوط به ورودی اشتباه کاربر چک شده و هم خطاهایی نظیر divide by zero و همچنین همه توابع با بلاک های try catch در برابر exception مقاوم شده اند و اگر جایی مشکلی بوجود آید در بلاک catch ما تابع closeEveryThing را صدا میزنیم تا منبعی به اشتباه هدر نرود.