--Opis implementacji

W moim projekcie zostały zaimplementowane następujące klasy:

1.Server - odpowiada za utworzenie potrzebnej illości socketów i przekazanie ich do wątków obsługujących żądania wysłane do tych socketów.

1.1 Otrzymuje liste socketów jako argumenty uruchomienia programu

1.2 Tworzy obiekty klasy DatagramSocket na podstawie przekazaych poprzednio portów.

1.3 W metodzie listenPorts() uruchamia wątki (ServerThread) które będą przetwarzali żądania wysłane na przekazane jako argument konstruktora obiektu ServerThread sockety.

2.ServerThread - obsługuje żądania wysłane na przekazany jako argument konstruktora tej klasy socket.

2.1 W pętli while oczekuje na odebranie datagramy z komunikatem "CON".

2.2 Po odebraniu datagramu z odpiwiednim komunikatem tworzy nowy socket i przekazuje sekwencje poprawnych portów,utworzony socket na kturym będzie odbywała komunikacja pomiędzy klientem a serwerem , datagram otrzymany od klienta,port na który serwer otrzymał datagram od użytkownika i końcowy plik który będzie transmitowany w przypadku wysłania pakietów w poprawnej kolejności jako argumenty konstruktora obiektów klasy OverloadConnectionThread.

3.OverloadConnectionThread - komunikuje z klientem na wskazanym poprzednio porcie i decyduje o wysłaniu pliku końcowego.

3.1 Po uruchomieniu metody execute() wysyła datagram o początku komunikacji na nowym porcie przy pomocy metody (3.4)absoluteSend() po wysłaniu potwierdzenia dodaje nowego użytkownika (jako instancja klasy User) na podtawie klucza (adres użytkownika+port użytkowniaka) albo dodaje nowy port do listy portów odwiedzonych przez użytkownika (User.confirmations).

3.2 Natępnie sprawdza czy można sprawdzić sekwencje odwiedzonych przez użytkownika portów w przypadku pozytywnego rezultatu sprawdza sekwencje.

3.3 Jeżeli sekwencja jest poprawną zostaje wyłowana metoda (3.5)shareFile() z utworżonym i przekazanym jako argument soketem do wysłania pliku.

3.4 Metoda - absoluteSend(DatagramSocket socket, DatagramPacket packet) polega na wysłaniu przekazanego jako argument komunikatu po czym oczekuje na potwierdzenie odebrania od odbiorcy "ACK" i wysyła komunikat "NORETR" o zatrymaniu oczekiwania na retransmisję po stronie odbiorcy.

3.5 Metoda - shareFile(DatagramSocket dataSocket) najpierw wysyła datagram z nazwą plika i jego rozmiarem przy pomocy metody (3.4)absoluteSend() następnie w pętli odzcytuje plik do buforu i wysyła odczytane dane przy pomocy (3.4)absoluteSend() po wysłaniu ostatniej datagramy z danymi pliku wysyła datagram przy pomocy metody (3.4)absoluteSend() z komunikatem "END" który sygnalizuje o końcu transmisiji pliku.

4.User - zapewnia przechowyłanie danych użytkownika i ich sprawdzenie.

4.1 Metoda - canBeChecked() sprawdza czy illości odwiedzonych przez użytkownika portów nie równa dozwolonej illości odwiedzonych portów.

4.2 Metoda - stop() zatrzymuje wykonanie wątku liczącego czas znajdowania na liście użytkowników.

4.3 Metoda - checkConSequence(String []right_sequence) sprawdza czy odpowiada sekwencja odwiedzonych portów przechowywana użytkownikiem sekwencji portów przekazanej jako argument.

5.Client - komunikuje z serwerem przez wskazane jako argument porty.

5.1 Uruchamia metode completeLogin() która polega na wysłaniu datagramu z komunikatem "CON" do serwera i oczekiwanie na odpowiedż z innego portu dla osobnej komunikacji z serwerem przy pomocy metody (5.2)absoluteReceive() następnie oczekuje na odebranie rezultatu zwrotu do serweru przy pomocy metody (5.2)absoluteReceive() (ACK albo nie otrzymamy nic w przypadku odmowy) po odebraniu odpowiedzi w przypadku zwracania do ostatniego porta w sekwencji zostaje wyłowana metoda (5.3)getFile().

5.2 Metoda - absoluteReceive(DatagramSocket socket,DatagramPacket dt) odbiera datagram i wysyła datagram z potwierdzeniem("ACK") do odbiorcy po czym oczekuje na retransmisję w przybatku timeout albo odebrania datagramu z komunikatem "NORETR" zatrymuje oczekiwanie na retransmisję.

5.3 Metoda - getFile() otrzymuje nazwę pliku i jego rozmiar przy pomocy (5.2)absoluteReceive() następnie tworzy plik do zapisu z taką samą nazwą i w pętli zaczyna odbierać przy pomocy (5.2)absoluteReceive() i zapisywać do końcowego pliku otrzymane z serwera dane w przypadku wystąpienia datagramu z komunikatem "END" zatrymuje odbieranie obcina dane z ostatniego pakietu i zapisuje do pliku końcowego

--Jak zainstalować

Dostatecznym jest sciąganie folderu z projektem również koniecznym jest instalacja Java 8 w przypadku jej braku i ustalenie Java\jdk1.8.0_201\bin jako zmiennej systemowej

--Jak uruchomić

Najpierw trzeba uruchomić plik compile.bat po uruchomieniu compile project zostanie skompilowany następnie trzeba otworzyć plik run(Server) i wpisać po (start java -cp src Server) numery portów (1024<port<65536) przez spacje następnie trzeba uruchomić ten plik po uruchomieniu pliku run(Server) trzeba otworzyć plik run(Client) i wpisać po (start java -cp src Client) adres servera oddzielony spacjamy po obu stronach (jeżeli server uruchomiony na tym samym komputerze wpisać 127.0.0.1 albo localhost) i numery portów (1024<port<65536) przez spacje następnie trzeba uruchomić ten plik.