Spis treści:

1 OPIS WĘZŁA

każdy węzeł posiada 3 pola:

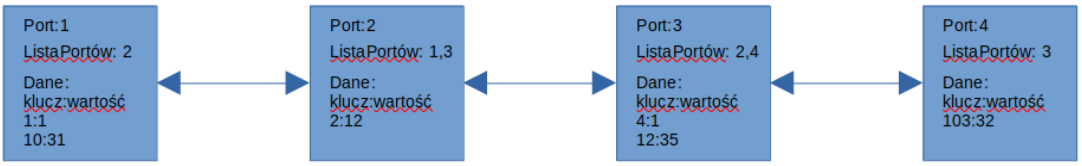
```
pole do przechowywania danych
HashMap<Integer, Integer> database;

pole do przechowywania informacji o tym z jakimi węzłami serwer jest połączony (jest w relacji )
private ArrayList<Integer> connectedPorts;

Pole do przechowywania własnego nr portu
private int portNumber;
```

2 SPOSÓB ORGANIZACJI SIECI

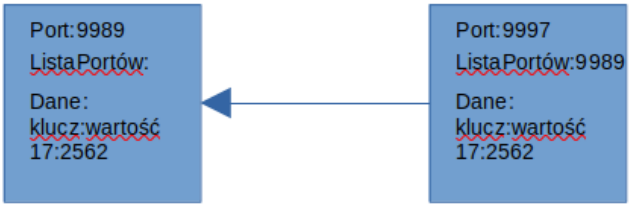
A. Sieć węzłów docelowo powinna być ustawiona szeregowo, to znaczy każdy z węzłów maksymalnie może posiadać informacje o dwóch węzłach sąsiadujących:

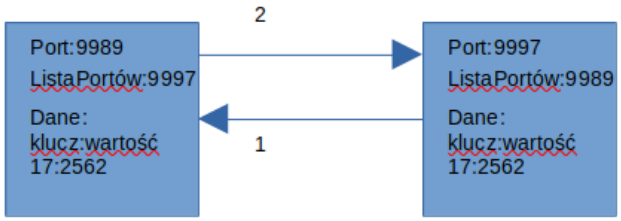


B. Dodawanie węzłów do sieci odbywa się za pomocą komend:
Poniższy przykład oznacza że na porcie 9989 stworzona została baza danych która posiada w sobie rekord 17:2562
java DatabaseNode -tcpport 9989 -record 17:2562

Dodanie kolejnego węzłą sieci:
Poniższy przykład oznacza, że na porcie 9997 stworzona została baza danych która posiada w sobie rekord 15:2564, oraz że połączyła się z portem 9989 za pomocom protokołu TCP

Czyli po utworzeniu węzła natychmiast jest wysyłany komunikat do 9989, dodaje do jego listy portów swój nr czyli w tym przypadku 9989, a ten odpowiada I też wysyła komunikat do 9997 I dodawany jest nr portu do listy
java DatabaseNode -tcpport 9997 -record 15:2564 -connect localhost:9989





w tym momencie oba węzły posiadają na listach portów swoich sąsiadów, dzięki czemu węzły sieci są ze sobą połączone (wiedzą gdzie jest następny port)

Operacja ta jest powtarzana kiedy tworzony jest nowy węzeł sieci np:
java DatabaseNode -tcpport 9990 -record 17:2563 -connect localhost:9997

WAŻNE!:

Operacje

java DatabaseNode -tcpport 9990 -record 17:2563 -connect localhost:9997 -connect localhost:9989
java DatabaseNode -tcpport 9991 -record 12:2562 -connect localhost:9990 -connect localhost:9997 -connect localhost:9989

Zadziałają natomiast będzie to błędne ze **sposobem organizacji sieci,** ponieważ to by oznaczało, że byłoby połączenie każdy z każdym, a przez to dalsze funkcjonalności nie działałyby poprawnie

3 OPIS PROTOKOŁU KOMUNIKACJI

1 klient-serwer: TCP

Zapewnia niezawodne, orientowane na połączenie strumienie danych. Wszystkie wysłane pakiety docierają do miejsca przeznaczenia w odpowiedniej kolejności.

Protokół klient-serwer: Gdzie klient inicjuje połączenie, wysyła żądanie i oczekuje na odpowiedź od serwera. Po Otrzymaniu odpowiedzi klient odłącza się od sieci.

2 serwer-serwer: TCP

Za pośrednictwem wątków takich jak `MaxValueThread`, `MinValueThread`, `GetValueThread`, czy `FindKeyThread`, każdy z nich otwiera nowe gniazdo TCP (`Socket`) do komunikacji z innym serwerem. Każdy z nich posiada swoje pola w których przekazuje informację dalej. Oraz pola takie jak nextServerPort, sender port, dzieki którym informacja wie z jakiego miejsca zostala wysłana I gdzie ma trafić, co zapobiega nieskończonym pętlom.

4 OPIS PRZESYŁANYCH KOMUNIKATÓW

możliwe komunikaty na **serwerze:**
Client connected - oznacza, że zostało nawiązane połączenie z węzłem, nie koniecznie oznacza, że jest to klient, może to być też inny węzeł w sieci

DatabaseNode is running on port <Nrportu> - Węzeł działa na porcie

<Nazwa_operacji>Thread started for port <nrportu>
Response from server
<Nazwa_operacji>Thread finished for port <nrportu> – te trzy komunikaty wyświetlane są praktycznie zawsze razem, oznacza to że informacja jest przekazywana między węzłami

Could not listen on port <nrportu> - oznacza że port jest zajęty
Node <nrportu> is terminating: - oznacza że wykonana została operacje terminate

możliwe komunikaty do **klienta:**
No connection with localhost. - oznacza, że nie jest możliwe połączenie zwykle jest to zpowodowane wpisaniem złego nr portu localhosta na którym ma być wykonywana operacji
ERROR – kiedy operacja nie została wykonana(zależy od operacji)
OK – Kiedy operacja się powiedzie
invalid operation argument – kiedy agrument dla operacji jest zły np dla set-value 1:w
Invalid command – kiedy jest źle wpisana komenda albo nie istnieje

5 OPIS PROCESU KLIENCKIEGO

możliwe komendy (wartości są przykładowe) **GŁÓWNE**

java DatabaseClient -gateway localhost:9989 -operation set-value 2:1000
java DatabaseClient -gateway localhost:9989 -operation get-value 2
java DatabaseClient -gateway localhost:9989 -operation find-key 2
java DatabaseClient -gateway localhost:9989 -operation get-max
java DatabaseClient -gateway localhost:9989 -operation get-min
java DatabaseClient -gateway localhost:9989 -operation new-record 13:1009
java DatabaseClient -gateway localhost:9989 -operation terminate

DODATKOWE

java DatabaseClient -gateway localhost:9992 -operation showDataBase
java DatabaseClient -gateway localhost:9991 -operation showConnectedPorts

Opis działania serwera dla operacji **set-value <klucz>:<wartość>:**

W sytuacji kiedy baza posiada **jeden** serwer na liście connectedPorts;

2 możliwości:

1. jeżeli szukana wartość set-value znajduje się od razu na serwerze z którym klient się połączył, zostanie zamieniona tylko ta wartość, I nie będzie wysyłany sygnał dalej do szukania.

2. jeżeli szukana wartość set-value **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostanie wysłany sygnał do następnego serwera na liście. Operacja powtarza się dla wszystkich serwerów w bazie do momentu aż nie zostanie znaleziony klucz. W takim przypadku zostanie zmieniona tylko **jedna** komórka klucz:wartość

W sytuacji kiedy baza posiada **dwa** serwery na liście connectedPorts;

2 możliwości:

1. taka sama jak w przypadku jednego serwera naliście

2. jeżeli szukana wartość set-value **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostaną wysłane DWA sygnały ponieważ są dwa serwery na liście. Przez co maksymalnie mogą się zmienić **dwie** wartości jeżeli klucze są takie same.

Komunikaty to OK lub ERROR

Opis działania serwera dla operacji **get-value <klucz>:**

W sytuacji kiedy baza posiada **jeden** serwer na liście connectedPorts;

2 możliwości:

1. jeżeli szukana wartość get-value znajduje się od razu na serwerze, z którym klient się połączył, zostanie pobrana tylko ta wartość, I nie będzie wysyłany sygnał dalej do szukania.
2. jeżeli szukana wartość set-value **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostanie wysłany sygnał do następnego serwera na liście. Operacja powtarza się dla wszystkich serwerów w bazie do momentu aż nie zostanie znaleziony klucz. W takim przypadku zostanie pobrana tylko jeden rekord klucz:wartość który jako pierwszy zostanie znaleziony

W sytuacji kiedy baza posiada **dwa** serwery na liście connectedPorts;

2 możliwości:

1.taka sama jak w przypadku jednego serwera naliście

2. jeżeli szukana wartość set-value **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostaną wysłane **DWA** sygnały ponieważ są dwa serwery na liście. Jeżeli klucze są takie same to I tak zostanie zwrócona jedna wartość, a mianowicie ta którą wątek/sygnał zwrócił jako pierwszy

Komunikaty to <klucz><wartość> lub ERROR

Opis działania serwera dla operacji **find-key <klucz>:**
W sytuacji kiedy baza posiada **jeden** serwer na liście connectedPorts;

2 możliwości:

1. jeżeli szukana wartość find-key znajduje się od razu na serwerze, z którym klient się połączył, zostanie pobrany tylko port, I nie będzie wysyłany sygnał dalej do szukania.

2. jeżeli szukana wartość find-key **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostanie wysłany sygnał do następnego serwera na liście. Operacja powtarza się dla wszystkich serwerów w bazie do momentu aż nie zostanie znaleziony klucz. W takim przypadku zostanie przekazny port który został znaleziony jako pierwszy

W sytuacji kiedy baza posiada **dwa** serwery na liście connectedPorts;

2 możliwości:

1. taka sama jak w przypadku jednego serwera naliście
2. jeżeli szukana wartość find-key **NIE** znajduje się od razu na serwerze z którym klient się połączył, zostaną wysłane **DWA** sygnały ponieważ są dwa serwert na liście. Ale I tak zostanie zwrócona jedna wartość portu, a mianowicie ta której wątek/sygnał wrócił z kluczem jako pierwsz. Jęzli klucze są takie same.

Komunikaty to <adres>:<wartość> lub ERROR

Opis działania serwera dla operacji **get-max/get-min:**
W sytuacji kiedy baza posiada **jeden** serwer na liście connectedPorts;

1 możliwości:

1. Szukana jest największa wartość w aktualnym węźle I jest przekazywana dalej do następnego serwera, operacja jest powtarzana dla wszystkich węzłów w sieci

W sytuacji kiedy baza posiada **dwa** serwery na liście connectedPorts;

1 możliwości:

2. Szukana jest największa wartość w aktualnym węźle I jest przekazywana dalej przez **dwa** sygnały/wątki do następnego serwera, operacja jest powtarzana dla wszystkich węzłów w sieci.

Komunikaty to <klucz><wartość>

Opis działania serwera dla operacji **new-record <klucz><wartość>:**

1 możliwości:

1. Na aktualnym serwerze do którego połączony jest klient dodawany jest nowy rekord

Opis działania serwera dla operacji **terminate:**

1 możliwości:

1. Serwer wysyła sygnał (do wątków które posiada na liście portów), którym jest nr swojego portu serwery otrzymują wiadomość I usuwają z listy wysłany port. Następnie serwer wysyła OK I po 500 milisekundach kończy działanie z komunikatem “**Node <nrportu> is terminating:**”