

Design

For the Client, I create Phase, Request, Utils, WriteFile and StartClient Class.

StartClient class is the main function for the whole project, it is an entry. This takes run arguments from the input like

```
public static int maxThreads = 0;
public static int numOfSkiers = 0;
public static int numOfSkierLift = 0;
public static int meanNumberOfSkiLift = 0;
public static String port = "http://localhost:8080/cs6650lab";
```

Phase class is the phase object that can start number of phases.

Request class is used in phase class to run the post request

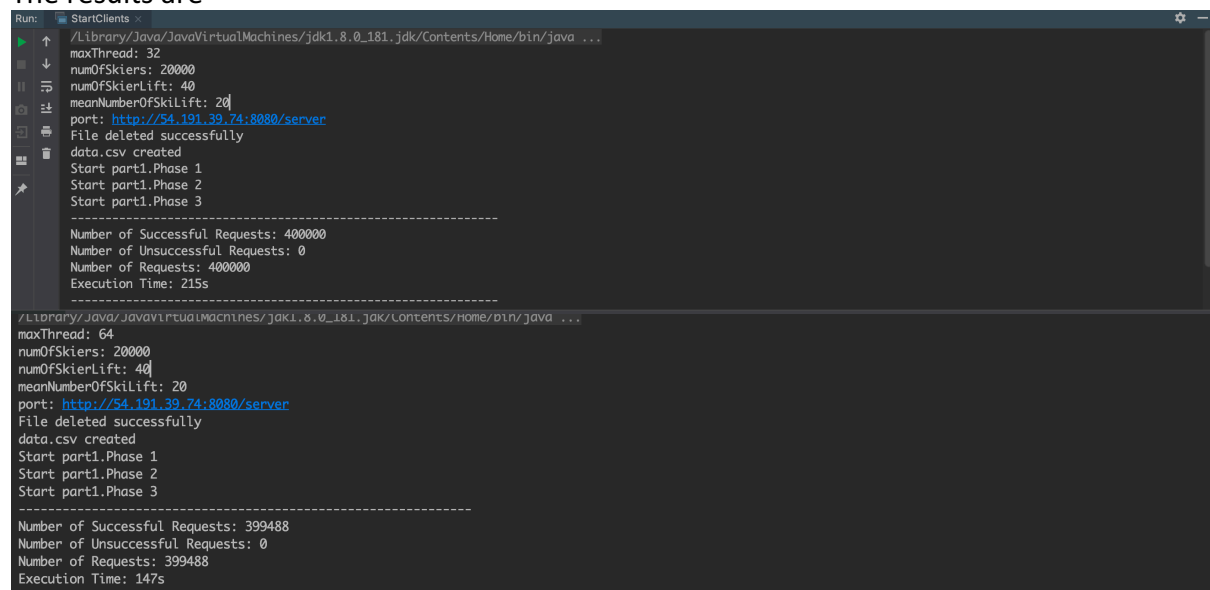
Utils class is for some helper function, like analyzing results and preparation.

WriteFile is another runnable object that can write request to the csv.file.

Basically, I use countdownlatch to count the 10% threads finish for phase 1 phase2. All the countdownlatches are initialized in the main function and passed into request class to do a countdown, when a request is finished. In this way, I can start phase2 and phase3 correctly. Use other countdownlatches to wait all threads finished so that we can analyze the results.

For part2 in separate folder, I use one second bucket to store the number of requests and total latency for one second. So that some requests' are in the same bucket. If the number of requests is huge and the programs will take a long time. The cache in the computer might overflow. This is no good. For the result, even though it is not that accurate in the end for the small number of requests, but it will have better performance for a large requests.

The results are



```
Run: StartClients
/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java ...
maxThread: 32
numOfSkiers: 20000
numOfSkierLift: 40
meanNumberOfSkiLift: 20
port: http://54.191.39.74:8080/server
File deleted successfully
data.csv created
Start part1.Phase 1
Start part1.Phase 2
Start part1.Phase 3
-----
Number of Successful Requests: 400000
Number of Unsuccessful Requests: 0
Number of Requests: 400000
Execution Time: 215s
-----

/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java ...
maxThread: 64
numOfSkiers: 20000
numOfSkierLift: 40
meanNumberOfSkiLift: 20
port: http://54.191.39.74:8080/server
File deleted successfully
data.csv created
Start part1.Phase 1
Start part1.Phase 2
Start part1.Phase 3
-----
Number of Successful Requests: 399488
Number of Unsuccessful Requests: 0
Number of Requests: 399488
Execution Time: 147s
-----
```

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java ...
maxThread: 128
numOfSkiers: 20000
numOfSkierLift: 40
meanNumberOfSkierLift: 20
port: http://54.191.39.74:8080/server
File deleted successfully
data.csv created
Start part1.Phase 1
Start part1.Phase 2
Start part1.Phase 3
-----
Number of Successful Requests: 399488
Number of Unsuccessful Requests: 0
Number of Requests: 399488
Execution Time: 76s
-----
/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java ...
maxThread: 256
numOfSkiers: 20000
numOfSkierLift: 40
meanNumberOfSkierLift: 20
port: http://54.191.39.74:8080/server
File deleted successfully
data.csv created
Start part1.Phase 1
Start part1.Phase 2
Start part1.Phase 3
-----
Number of Successful Requests: 399360
Number of Unsuccessful Requests: 0
Number of Requests: 399360
Execution Time: 48s
-----
```