Manual

Link to video demonstrating system

https://www.youtube.com/watch?v=XPkY5BExC40

Central

To start a central, run the Central.jar file from a console.

```
java –jar Central.jar
```

Peer

To start a peer, run the Peer.jar file from a console with the IP of the central as program argument. The peer will automatically join on the specified central. Note that only one peer can be joined on a central for each IP.

```
java –jar Peer.jar <Central ip>
```

In some cases it might be necessary to add VM options: -Djava.rmi.server.hostname=<own IP4 address>

Researcher

<u>Default</u>

To start the default researcher, run the Researcher.jar file from a console with the IP of the central as program argument. The researcher will automatically join on the specified central. Note that only one researcher can be joined on a central for each IP.

```
java –jar Researcher.jar <Central ip>
```

In some cases it might be necessary to add VM options: -Djava.rmi.server.hostname=<own IP4 address>

To start a matrix multiplication calculation (800x800x50), write "startMatrixCalc" in the console.

To start a Collatz calculation (in the interval 0-10,000,000), write "startCollatzCalc" in the console.

Defining a problem

To define your own problem, first define the kind of task to be executed by writing a class that implements the Task interface. The task should have the method execute that executes the task and returns the result.

Write a class that implements the interface SchedulingProblem.

- The init method should initiate the problem calculation. Specifically it must create and fill a task queue with the tasks to be executed.
- The getAmountOfNeededPeers method should return the amount of peers needed to perform the calculation.
- The getAvailableTasks must return the task queue containing tasks to be executed at any given time.
- The handleTaskResult takes a task and the result of that task after being executed and handles that result.
- The isDone method should return true if the problem has been solved and false otherwise.
- The compare method takes two tasks and return whether the two tasks are equal.
- The reset method resets everything that needs to be reset in order to start over a calculation.

The problem definition can be used by modifying the DefaultResearcherRunner class so that the scheduler is created from the problem definition.