Name: Daryan Chan

ID: 113973192

Section: NAA

Task 1

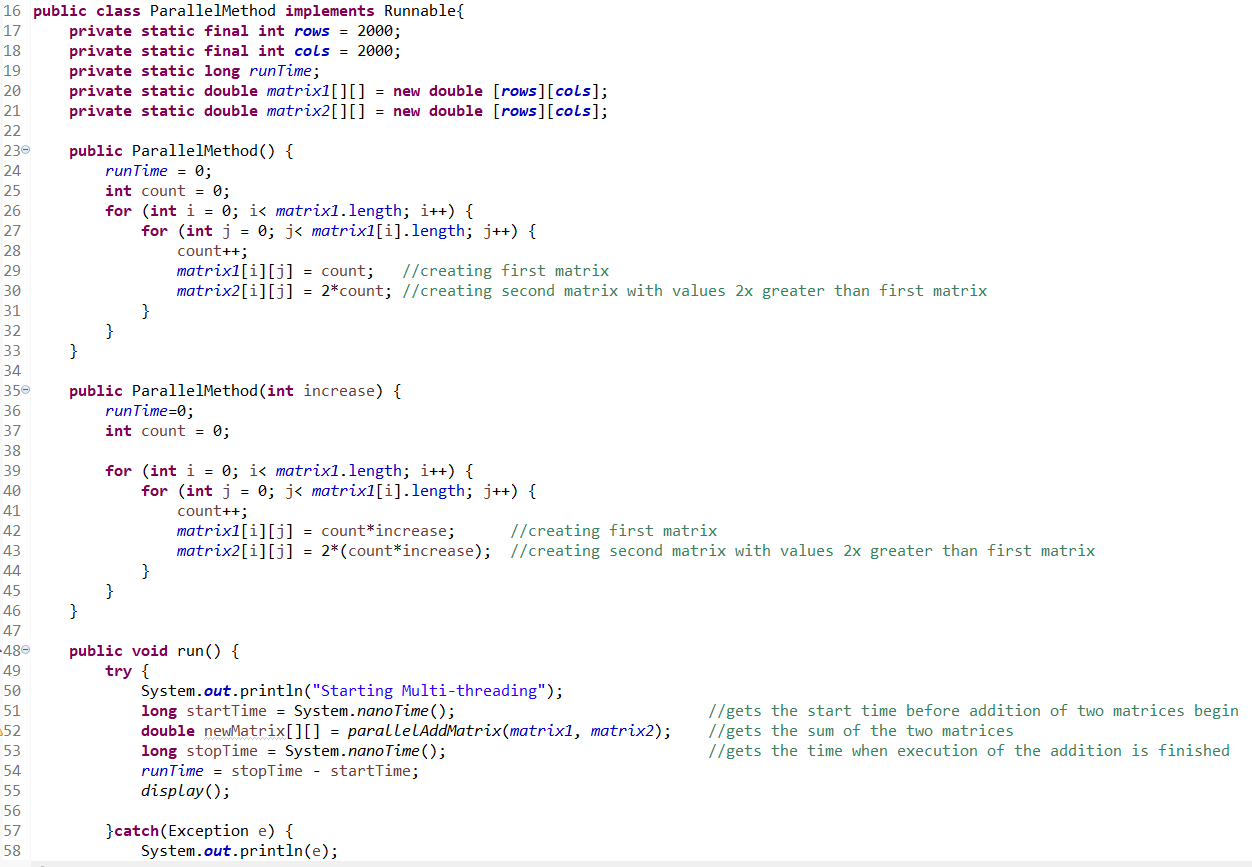


Figure ParallelMethod class

For this class, I created the constructor for the purpose of initializing my variables at the moment of creation.

The run method starts when the start() begins. This will use a try catch method and will print out the message as well as record the start and end time of the execution. Then it will call the display function.

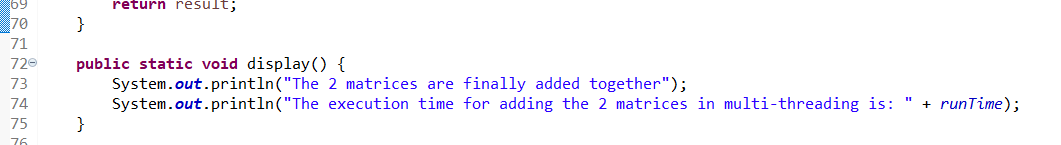


Figure ParallelMethod continuation

The display method will display the following messages and show the total run time.

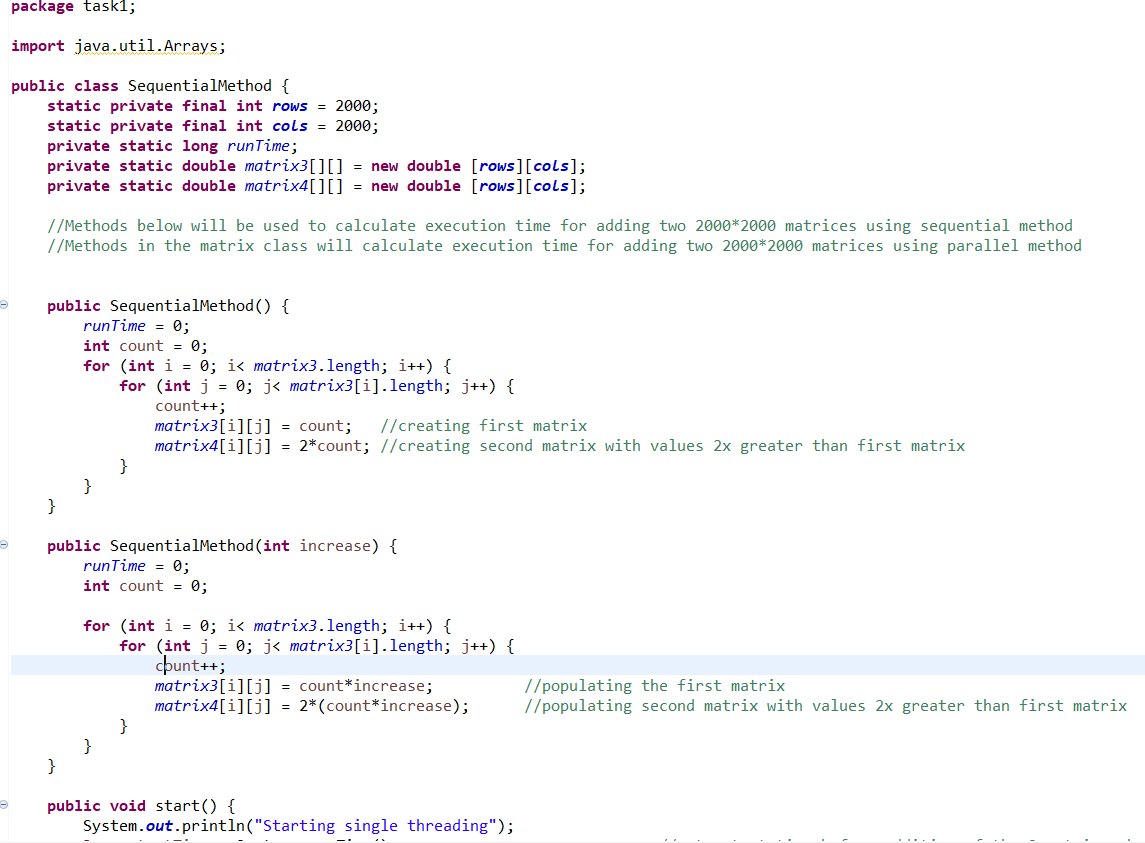


Figure SequentialMethod class

This class shows the constructors and its purpose is to initialize the variables.

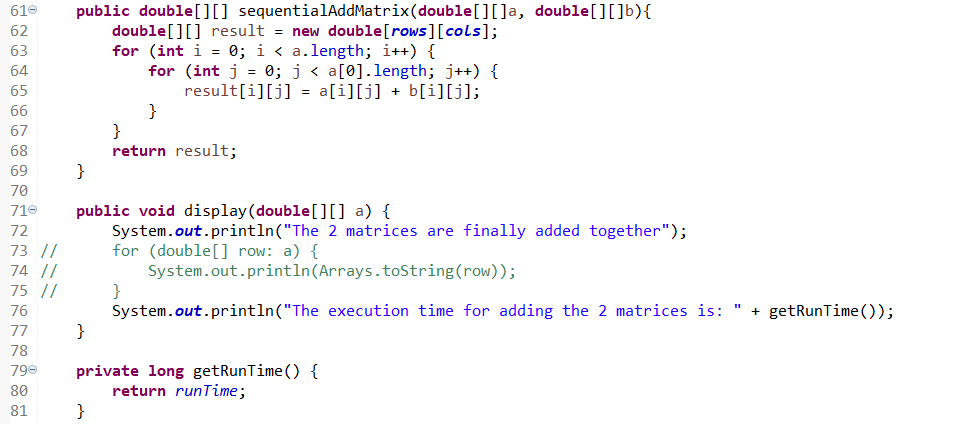


Figure SequentialMethod class continuation

This shows the add matrix, which basically goes through a for loop 2 times so that each row and column of each matrix is added together. The display method will display the messages and show the run time.

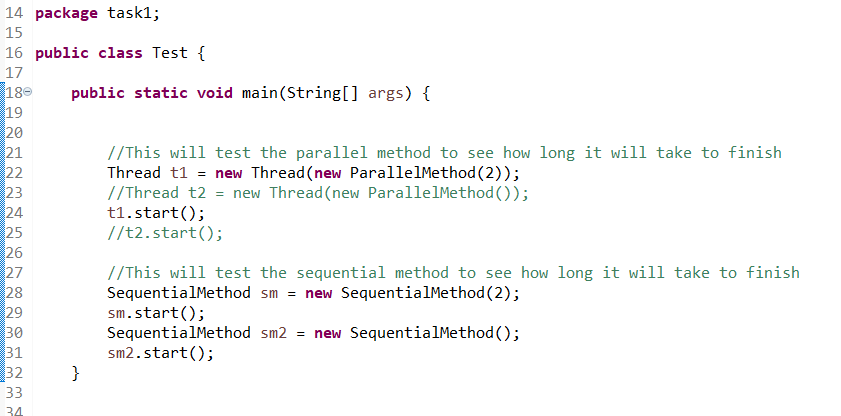


Figure Test class

This will create all the objects and creates the multiple threads and start them

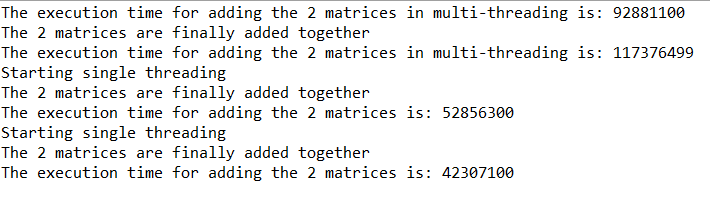


Figure Results

The run time is shown

Task 2

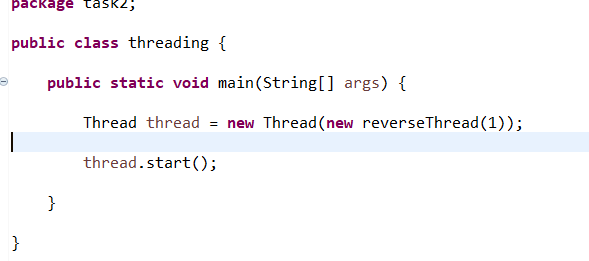


Figure threading class

This is the main method and will create a thread and start it, using reverseThread as an object.

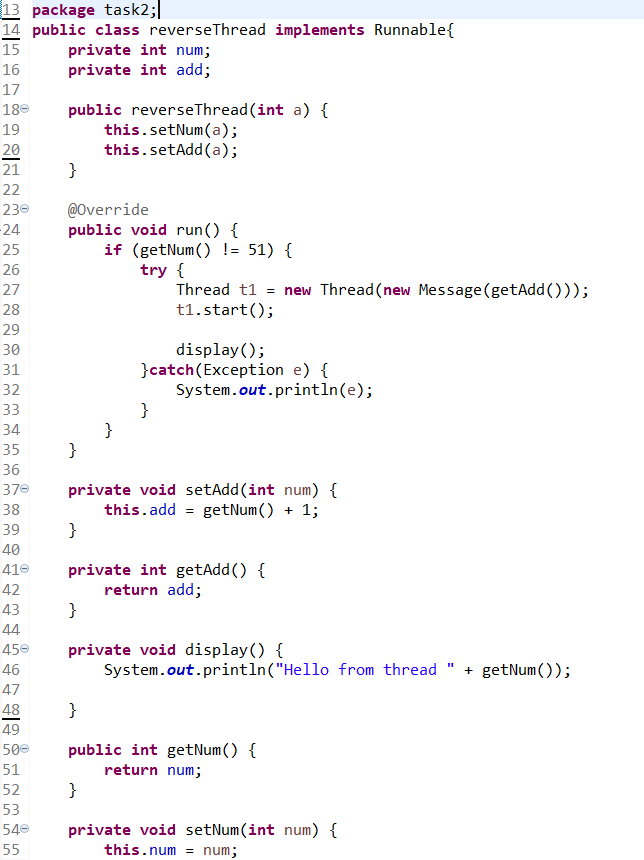


Figure reverseThread class

This class implements runnable and has a constructor. This will initialize the variables. It also has run method that will begin when the start() method is called in the main method. The display method will display the message and shows the thread number. The rests are getter and setters functions.

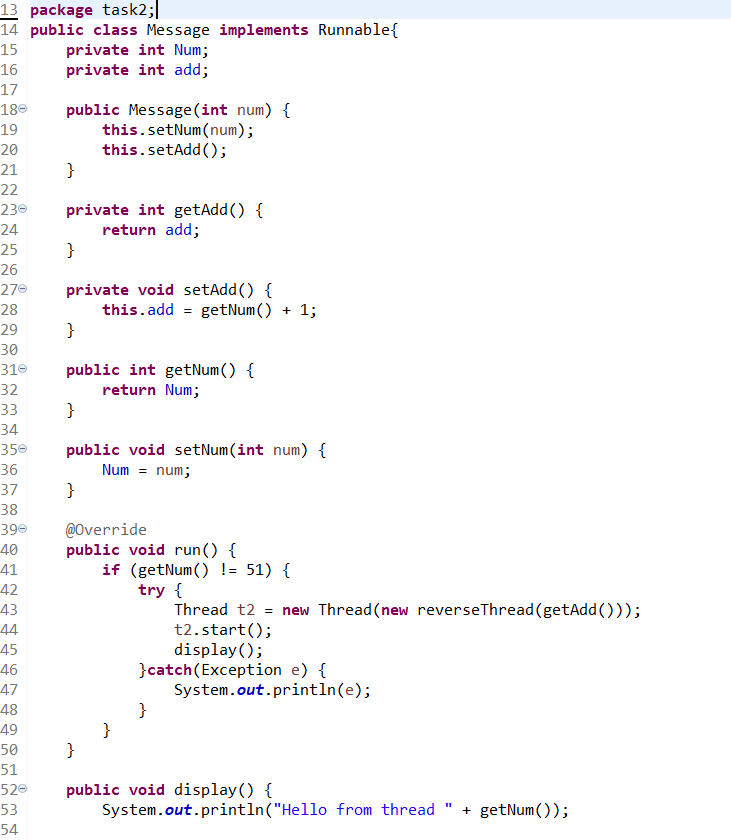


Figure Message class

The idea behind this class is the same as the previous page. Only difference is the run method. For this, it will create a new thread using reverseThread as the object and the idea behind this is that the threads will create other threads.

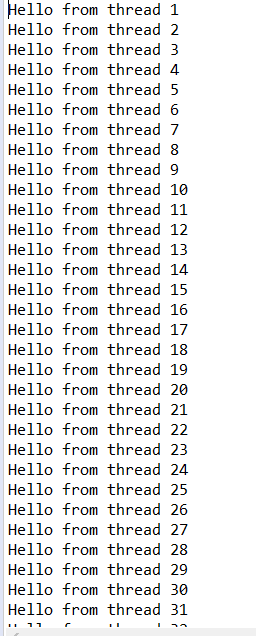


Figure Results

This goes all the way to thread 50.