

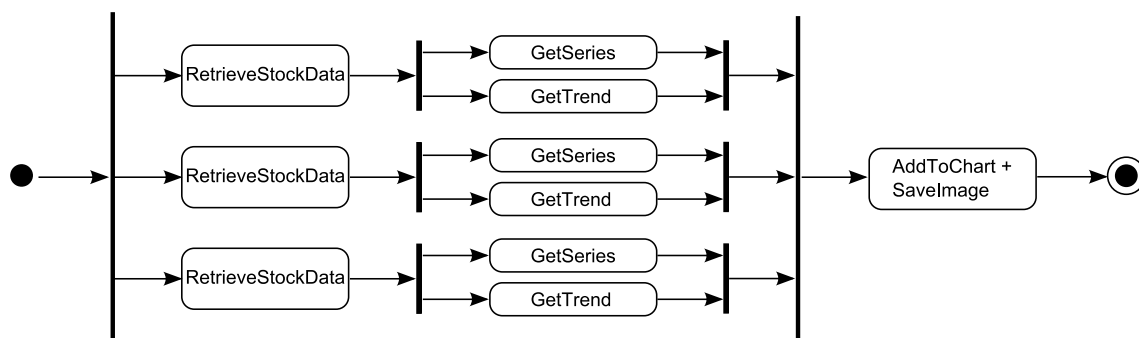
Name _____

Points _____ Effort in hours _____

1. Stock Data Visualization**(6 + 6 Points)**

Web requests can often be executed asynchronously to improve the responsiveness of an application and to reduce the total response time of multiple requests. On Moodle we provide a sequential implementation of a stock data visualization program that downloads price histories of three stocks from *quandl.com* and shows them in a line chart.

- a) Implement an asynchronous version of the stock data visualization program using the .NET Task Parallel Library. Your version should execute tasks as shown in the following activity diagram. Use continuations to create chains of several tasks.



- b) Implement a second version of the stock data visualization program which uses the keywords *async* and *await*. Make sure the tasks are again modeled as shown in the activity diagram.

2. Parallel Water**(8 + 4 Points)**

Implement a parallel version of the Water simulation from the first exercise using the .NET Task Parallel Library. You can parallelize either your already optimized version or the original version provided on Moodle.

- a) Think about how you can separate data and work of the Water simulation to allow parallel processing and consider appropriate synchronization. Describe and implement your decomposition and parallelization concept and explain your approach and its advantages and maybe also disadvantages.
- b) Test your parallel Water simulation, compare its runtime with your optimized version and the original version from Exercise 1 and calculate its speedup.