COSC2P13 Group Project

Part 1

- Parallelism
  - Used synchronized to prevent full parallelism of the threads
- Used threads to do calculations and add them all up at the end
- Inherent issue
  - Deadlock prone
  - Solution
    - Locking 2 variables at the same time

Part 2

- Deadlock safe because it implements two strategies to prevent deadlocks
- Strategy 1
  - Lock variables copied to a separate object before they are locked
- Strategy 2
  - Even threads take the left ID first, odd threads take the current ID first
- Parallelism is reduced because threads may be waiting for locks to be unlocked

Part 3

- How much overhead has file logging added to the performance of your program?
  - File logging added a lot of delay because of the way we implemented it
    - Would reopen file for each write.
  - Runtime was SERVERLY slower than it was without logging.
- How have you relieved the I/O delay to improve performance?
  - Had to make it faster, because it would grow linearly with the number of terms being calculated.
  - Used a BufferedWriter to keep changes in memory, only writing when there was a lot to write in the buffer
    - This reduced the number of writes required, meaning there was almost no overhead anymore.

Part 4 (Server and Client)

- How fat is your client?
  - The client is fairly thin, all of the processing is handled by the server.
    - The client requests information and the server processes it.
- What information have you transferred between client and server?
  - The server sends over questions requesting input. The client inputs the appropriate information and sends it. The server then sends the processed information and the client logs it.
    - We settle for using PrintStream for the communication process as BufferedWriter didn't work.
- Have you used reliable or unreliable communication between your client and server? Why?
  - We used unreliable communication. It reduces overhead and complexity.
    - Continuous testing, allowed us to be sure this method would work. Reliable was not needed.