**TimeLine (Tentative)**

|  |  |  |
| --- | --- | --- |
| **Week #** | **Class Deadlines** | **Project Deadlines** |
| Week 4 |  | Read Papers  Set up Environments   * Clion * GoogleTest * Install mingw |
| Week 5 | HW 1 Due | Meet and Propose High level algorithms  Start Sequential Projects   * BST’s : Bobby * AVL : Tyler |
| Week 6 |  | Finish Sequential  Unit Tests for Sequential   * BST : Bobby * AVL : Tyler * Start Concurrent Data Structures |
| Week 7 |  | Finish Concurrent  Unit Tests for Concurrent   * BST : Tyler * AVL : Bobby |
| Week 8 | Exam 1 | * Start Report (Check again for what is needed) |
| Week 9 | Midterm Report due 10/17 | Finish Concurrent Data Structure Report   * BST : Bobby * AVL : Tyler |
| Week 10 |  | Implement Transactional Data Structures   * BST :Bobby * AVL : Tyler |
| Week 11 |  | Documentation   * BST : Bobby * AVL : Tyler |
| Week 12 |  | Construct Tests for Running Analsysis   * BST : Bobby * AVL : Tyler   Typically use a bash script and run through terminal |
| Week 13 |  | Finish Tests  Construct Graphs  Begin Presentation |
| Week 14 | Presentation DAY! | Finish Report |
| Week 15 | Final Report due 11/30 | Finish Report |

**What is Needed**

* C++
* Clion editor
* mingw with pthreads
* Gootle Test

**Part 1**

1. Implement Concurrent Data Structure
   * Internal BST
   * External BST
   * AVL Tree
2. Algorithms
   * PaVT – Traverse (key)
   * UpdateSpans( node)
   * Insert (key)
   * Remove (key)
3. Options for Tree Design
   * **3 classes (we will use this one)**
     + All 4 main algorithms will be implemented in each class
     + 12 algorithms total
     + Pros : Simple to write
     + Cons : Possibly reused code is frequent
   * Parent class Design (Interfaces)
     + All inherit from one class
     + Override algorithms when needed
     + Pros : Good design Practice
     + Cons : May not be easily implemented, will take long time, may not be possible
4. For Report
   * Progress Guarantee
   * Correctness Guarantee
   * Syncrhonization Techniques
   * How to improve the papers design
     + java 🡪 C++
   * Advantages and disadvantages compared to alternatives
     + This is a literature review
     + motivation for paper
     + Intro to our paper
     + Can be found in intro to authors paper
   * Improvements to our design
     + Object oriented design
     + Continue later
   * Obstacles in design
     + Memory management will be tricky

**Part 2**

1. Concurrent to Transactional
   1. Lock, atomic, volatile 🡪 RSTM
2. Report
   1. Compare concurrent data structure with RTSM with transaction size 1
   2. explain performance
   3. analyze increasing transaction size
   4. MODIFY TRANSATIONAL DATA STRUCTURE TO IMPROVE PERFORMANCE YAY
   5. Adress the following
      1. What modification did you implement
      2. Did they improve performance
      3. Include interpretation of results
      4. provide informal proof of correctness of modified implementation
3. Performance Evaluation
   1. Test Module
   2. Each thread executes
      1. fixed number of operations
      2. Vary distribution of operations
         1. Will use benchmark evaluations in paper
   3. Vary transaction size
   4. Plot 9 graphs
      1. 3 for each distribution of operations
      2. 3 for transaction size
         1. 3 x 3 = 9 combination of the former
      3. include interpretation of observed results