The code ia simulation for a Tomasulo algorithm, a method used in CPUs to dynamically schedule instructions to improve performance. Here's an outline of the report:

1. Introduction
- Purpose: Explain the aim of the code, which is to simulate the Tomasulo algorithm for instruction scheduling in processors.
- Scope: Outline the components involved - load/store buffers, reservation stations, register file, and instruction processing.

2. Code Structure and Components
- **Class Overview**: Describe the `Tomasulo` class and its role in the simulation.
- **Data Structures**:
  - LoadStoreBuffer: Used to manage load and store operations.
  - ReservationStation: Manages arithmetic and logical operations.
  - Register: Represents CPU registers and their statuses.
  - Instruction: Represents a CPU instruction and its various stages.
- **Main Methods**:
  - `initializeSimulation()`: Initializes simulation parameters.
  - `issue()`: Issues instructions to the appropriate buffers or stations.
  - `execute()`: Simulates the execution of instructions.
  - `writeBack()`: Handles the write-back phase of the instructions.
  - `advanceSimulation()`: Advances the simulation by one cycle.

The other classes such add and sub stations instruction and instruction status which has rd,rs,rt and not forget the fetch class the reads the lines
3. Approach and Algorithm
- **Tomasulo Algorithm**: Briefly explain the Tomasulo algorithm's principles.
- **Simulation Flow**: Discuss the flow of simulation from instruction issue to execution and write-back.

4. Testing Strategy
We Tested the code in practive assignment and lecture and tried to handle edge cases