CMSC 330: Organization of Programming Languages

Project 5
Multithreaded Metro Simulation

Goals

1. Implement simulation display
   • Examine log file of simulation events
   • Display state of simulation
2. Implement simulation verifier
   • Examine log file of simulation events
   • Discover illegal / missing simulation events
3. Implement multithreaded simulation
   • Separate threads for trains, passengers
   • Use synchronization to avoid data races
   • Use wait / notify for efficiency

Metro Simulation

You are given
• List of metro lines & stations on each line
• List of passengers & their stops
• Parser for reading simulation parameters / events
• Code for printing simulation events

You need to simulate
• Trains moving along metro line
• Passengers boarding / exiting trains

Simulation Parameters

Example

```plaintext
=== Lines ===
Red, Glenmont, Silver Spring, Bethesda

=== Trains ===
Red=1

=== Passengers ===
Amy, Silver Spring, Bethesda

=== Output ===
```

Simulation Events

Format

• Train <color, #> entering <station>
• Train <color, #> leaving <station>
• <Passenger> boarding train <color, #> at <station>
• <Passenger> leaving train <color, #> at <station>

Simulation Parameters

Format

```plaintext
=== Lines ===
<color>, <station 1>, <station 2>...

=== Trains ===
<color>=<num>

=== Passengers ===
<name>, <station 1>, <station 2>...

=== Output ===
<event>
```
Simulation Events

- Example
  - Train Green 1 leaving Fort Totten
  - Train Blue 1 entering L'Enfant Plaza
  - Train Red 1 entering Fort Totten
  - Train Yellow 1 entering Pentagon
  - Paul boarding train Yellow 1 at Pentagon
  - Train Green 1 entering Gallery Place
  - Train Red 1 leaving Fort Totten
  - Train Blue 1 leaving L'Enfant Plaza
  - Train Yellow 1 leaving Pentagon
  - Train Green 1 leaving Gallery Place

Simulation Display

- Example
  - Red
    - Glenmont
    - Silver Spring
    - Bethesda
    - [Red 2 Ann]
    - Amy
    - [Red 1]

Metro Simulation

- Multithreading
  - One thread per train
  - One thread per passenger

- Synchronization
  - Single train (from metro line) at station at a time
  - Passengers only board / exit when train is in station
  - Use enough locks to permit concurrent execution
  - Use wait / notify to avoid busy waiting

Simulation Rules

- Trains
  - Start by entering 1st station in metro line
  - Travel back and forth between 1st and last station
  - Stopping at all metro stations on line in order
  - For each metro line
    - May have multiple trains
    - Only one train in station at a time (regardless of travel direction)
    - Trains from different metro lines may be at same station
  - If no passengers in simulation
    - Each train must make at least 1 round trip from 1st station to last and back to 1st station (i.e., enter & exit 1st station twice)
    - Train may continue running after round trip

- Simulation completes
  - When all passengers reach destinations
    - Trains are allowed to continue moving a bit more
  - If no passengers in simulation
    - Each train must make at least 1 round trip
    - Go from 1st station to last station and back to 1st station

- Simulation Rules

  - Passengers
    - Start at 1st station on list of stops
    - Board & leave trains only when train is at station
    - Possible to miss train
      - Take future train
    - May board trains going in either direction
    - May change metro lines
      - If multiple lines at station