Assessment Project :- Railway Crossing Status Algorithm

step:1 Initialize crossingStatus to “CLOSED".

step:2 Loop:

a. Check for the presence of a train approaching the crossing:
- If a train is detected:

i. Set crossingStatus to "CLOSING" (to initiate the closing procedure).
ii. Start a timer for the closing procedure (e.g., 30 seconds).

b. If crossingStatus is “CLOSING":

i. Check the timer:

-If the timer expires:

- Set crossingStatus to “CLOSED”.
- Activate barriers and warning signals.

-Otherwise, continue with the closing procedure.

c. If crossingStatus is “CLOSED":

- Check for the absence of a train:
- If no train is detected:

- Set crossingStatus to "OPENING" (to initiate the opening procedure).

d. If crossingStatus is “OPENING":

i. Check the timer:

- If the timer expires:

- Set crossingStatus to “OPEN".

- Deactivate barriers and warning signals.

- Otherwise, continue with the opening procedure.

e. If crossingStatus is “OPEN”:

- Continue to monitor for an approaching train.To create a simple railway crossing status algorithm in Java, you can use a basic example with two states: "Open" and "Closed." You can simulate the railway crossing status change using a timer or a user-triggered event.

- We have a `RailwayCrossing` class that has a `boolean` field to represent whether the crossing is open or closed.

- The `open()` and `close()` methods change the status and print a message indicating the change.

- In the `RailwayCrossingSimulation` class, we create an instance of `RailwayCrossing` and use a `Timer` to simulate the change in the status every

- seconds. The `TimerTask` periodically toggles the status of the railway crossing.

Compile and run this Java program, and you'll see the railway crossing status changing between "OPEN" and "CLOSED" every 5 seconds. You can modify the timing and logic to suit your requirements.