1. Group Members:
    a. Kunal Shah(kgshah)
    b. Aadhav Sivakumar(asivakumar)
2. Behavior During TIME_STEP
    a. Calculate what velocity to spin each wheel at depending on the ground sensors
    b. Update odometry variables for this run
    c. Check for the start line. If we are at the start line then reset all vars to 0 and print out the error.
3. What happens when the robot's time step is not exactly TIME_STEP long, but slightly varies?
    a. These inconsistencies can occur in the real world especially if the code runs too slowly to keep up. When they occur then the odometry calculation can miss aggregating its step correctly or at all. Not to mention possibly missing a turn and the robot going off the line
4. ePuck's average speed (in m/s)
    a. 0.125m/s
5. In an ideal world, what should the ePuck's pose show each time it crosses the starting line?
    a. (0,0,0)
    b. aka: x=0,y=0 and θ=0
6. How did you implement loop closure in your controller?
    a. I checked for all ground sensors being under a threshold of the black line and I had a flag to make sure that the robot does not trigger many times on the same line and we were roughly at (0,0,0)
7. Time spent programming
    a. 4-hr line follower
    b. 2-hr Odometry
    c. 1-hr loop closure
8. Does your implementation work as expected?
    a. It works exactly as expected as we get a 1.8% error. The only difficulties we had was the line follower program jumping the line and detecting the start line(both detecting it exactly once and preventing false detections).