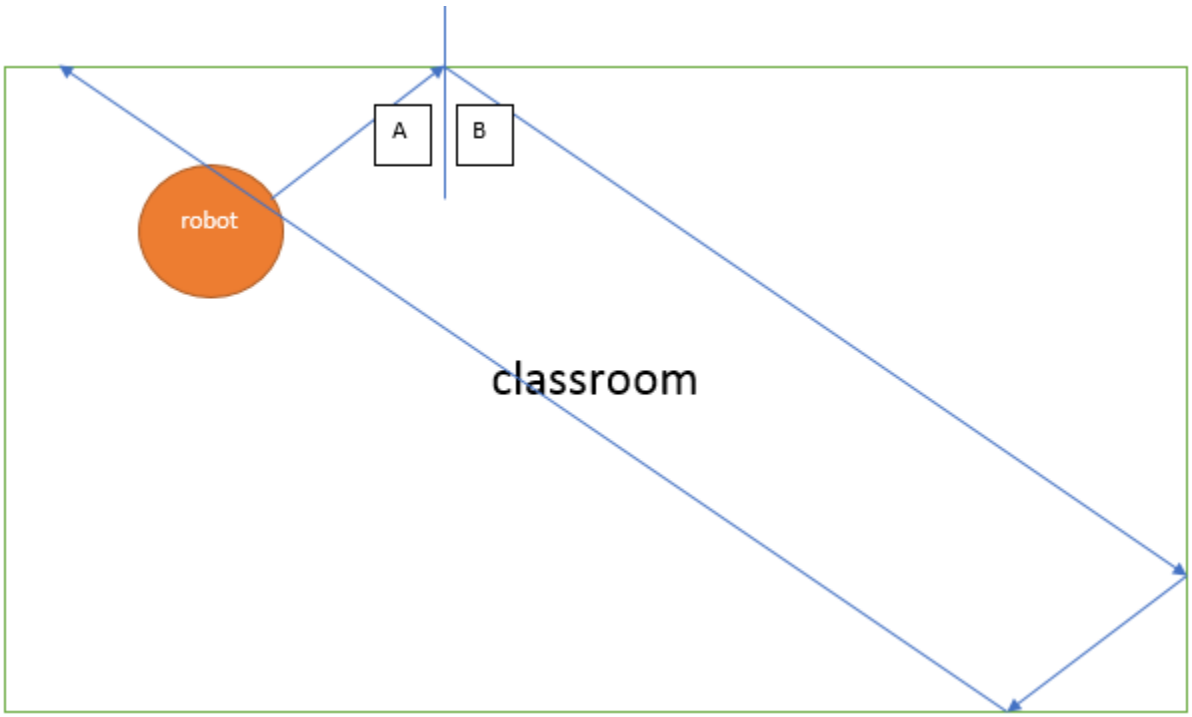


Problem statement:

The classroom has the size of L meter * W meter. The robot is initially at the position (x_0, y_0), and has an initial velocity vector of <a, b>. Find the number of bounce between the robot and the walls in the first 1000 seconds? What about in the first 100000000 seconds? (When the robot bounces with the wall, the velocity direction changes but the magnitude doesn't change. angle A = angle B)

1000 seconds (Will's idea):

Input(time:=1000)

For time: 1 → 1000

Position := Position + (vector)Current_Direction

If(out of bound)

Bounce# := Bounce# + 1

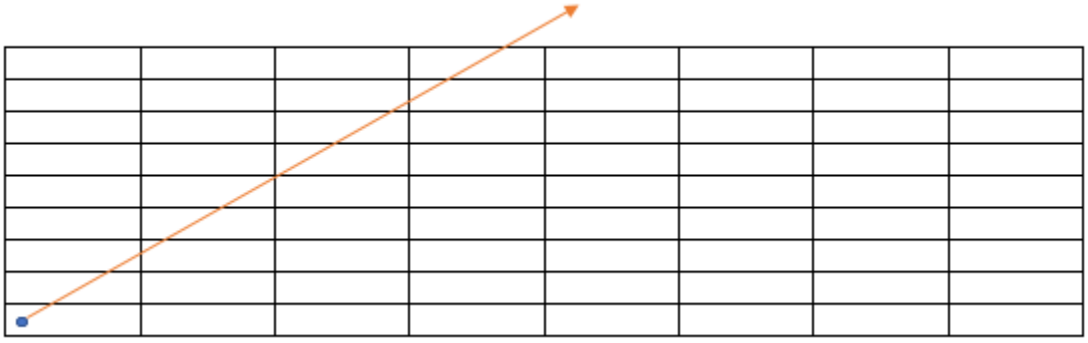
Change Current_Direction

Find the new Position

Output(Bounce#)

Estimated time complexity: O(N) N is the time the robot runs

100000000 seconds (Raymond's idea)



We can let the robot go and ignore the bounces. The bounces only cause the direction to change, and they don't change the magnitude of the velocity. In another word, because of symmetries, every time when the line crosses a segment, there's a bounce.

Estimated time complexity: O(1)