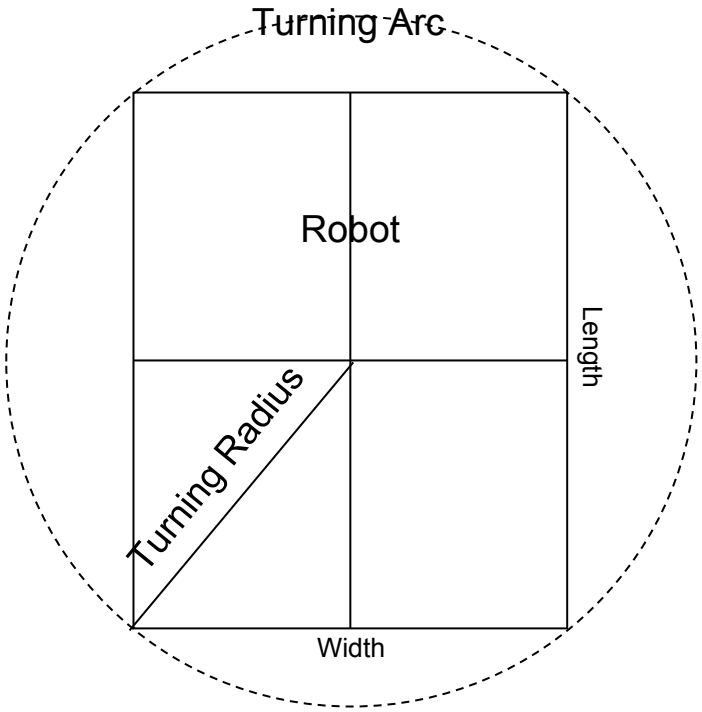

rot = 1 rotation
rev = 1 revolution
gr = gear ratio
wC = wheel circumference
tC = turning circumference
wr = wheel radius
tr = turning radius
W = width
L = length

All measurements are in inches

Ticks/inch: ticks_per_rot/rev * gr * rev/rC
Length of arc: degrees/360 * C
Pythagorean theorem: $C = \sqrt{(A^2 + B^2)}$

tr = pythagorean theorem; A ➡ W/2, b ➡ L/2
tC = 2π * tr

Inches to turn for *x* degrees = x/360 * tC
To turn:
- multiply by use ticks/inch formula, then divide by two
- turn one side that many inches positive
- turn the other side that many inches negative
- mot.setTargetPosition(mot.getCurrentPositon() ± …);
- if you used a double anywhere in you equation, cast the entire thing into an integer

Turn 90 degrees clockwise:

Robot width ➡ 14
Robot length ➡ 16
Wheel radius ➡ 2
Ticks per rotation ➡ 1440
Gear ratio ➡ 60

```
DcMotor L = hardwareMap.get(DcMotor.class, “left”);
DcMotor R = hardwareMapt.get(DcMotor.class, “right”);
L.setTargetPosition(left.getCurrentPosition() + (1440/1 * 60 * 1/(2 * Math.PI * 2)) * (90/360 * (2 * Math.PI * (Math.sqrt(Math.pow(14, 2) + Math.pow(16, 2))))));
R.setTargetPosition(right.getCurrentPosition() - (1440/1 * 60 * 1/(2 * Math.PI * 2)) * (90/360 * (2 * Math.PI * (Math.sqrt(Math.pow(14, 2) + Math.pow(16, 2))))));
```

I HIGHLY, HIGHLY recommend using variables for each step and keeping the setTargetPostion to getCurrentPosition ± turn_ticks:
ticks_per_rot, gear_ratio, robot_length, robot_width, wheel_radius, ticks_per_inch, etc.