moveCircle(int diam, int dir)

* initialize speed(v) to 10 // the default center linear speed is 10 (units TBD)
* initialize radius(r) to diam/2
* initialize rotational speed(w) to v/r
* initialize runtime(t) to 2\*pi/w
* If the direction is positive (counter clockwise):
  + find inner wheel linear speed from w: vinner = w \* (r-dleft) <- dleft is distance form center bot to left wheel and vleft is linear speed of left wheel
  + find outer wheel linear speed from w: vouter = w\*(r+dright) <- dright is distance form center bot to left wheel and vright is linear speed of left wheel
  + Set left motor speed to vinner speed
  + Set right motor speed to vouter speed
* Else direction is negative (clockwise):
  + find inner wheel linear speed from w: vinner = w \* (r-dright)
  + find outer wheel linear speed from w: vouter = w\*(r+dleft)
  + Set left motor speed to vouter speed
  + Set right motor speed to vinner speed
* Delay for t time
* Stop both wheels

moveFigure8(int diam)

* Run moveCircle with diam and positive direction
* Run moveCircle with diam and negative direction

go-to-angle(int angle)

* initialize speed(v) to 10
* initialize radius(r) to dleft(or d right) dleft is distance form center bot
* initialize rotational speed(w) to v/r
* initialize runtime(t) to (angle/360)\*2\*pi/w
* If the angle is positive:
  + Run Spinleft at speed 10
  + Delay for t time
  + Stop both wheels
* Else angle is negative:
  + Run Spinright at speed 10
  + Delay for t time
  + Stop both wheels

go-to-goal(int x, int y)

* initialize speed(v) to 10
* Initialize thetad(thetad) to atan2(y,x)
* Initialize distance(d) to sqrt(y^2+x^2)
* initialize runtime(t) to d/v
* Run go-to-angle(thetad)
* Run forward at speed v
* Delay for t time
* Stop both wheels

moveSquare(int side)

* initialize speed(v) to 10
* Run go-to-goal(side, 0)
* Delay for 1s
* Run go-to-goal(0, side)
* Delay for 1s
* Run go-to-goal(0, side)
* Delay for 1s
* Run go-to-goal(0, side)
* Delay for 1s

[LEONARDO, the Bipedal Robot, Can Ride a Skateboard and Walk a Slackline | www.caltech.edu](https://www.caltech.edu/about/news/leonardo-the-bipedal-robot-can-ride-a-skateboard-and-walk-a-slackline)

**Simulation:**

#include <Servo.h>

Servo leftservo;

Servo rightservo;

const int dleft = 50;

const int dright = 50;

void setup() {

  leftservo.attach(9);

  rightservo.attach(10);

  //move forward fast

  leftservo.write(170);

  rightservo.write(10);

  delay(5000);

  // circle

  circle(100,1); //works better for numbers closer to 100

  // figure 8

  figure8(60);

  // square

  square(1);

  //go to goal

  gotogoal(-5,-1);

}

void circle(int dia, int dir){

  int v = 100;

  int r = dia/2;

  float w = v/r;

  int t = (0.1858\*dia+3.2542)\*1000; //number was found by trendline assuming linear

  if (dir > 0){

    int vinner = map(w\*(r-dleft),0,v,90,180);

    int vouter = map(w\*(r+dright),0,v,90,0);

    leftservo.write(vinner);

    rightservo.write(vouter);

  } else if(dir < 0){

    int vinner = map(w\*(r-dright),0,v,90,0);

    int vouter = map(w\*(r+dleft),0,v,90,180);

    rightservo.write(vinner);

    leftservo.write(vouter);

  }

  delay(t);

  stop();

}

void figure8(int dia){

  circle(dia,1);

  circle(dia,-1);

}

void square(int l){

  for (int i = 0; i < 4; i++){

    leftservo.write(170);

    rightservo.write(-180);

    delay(1000\*l);

    gotoangle(3.1415/2);

  }

  stop();

}

void gotogoal(int x, int y){

  float theta = atan2(y,x);

  int l = sqrt(sq(x)+sq(y));

  gotoangle(theta);

  delay(1000);

  leftservo.write(170);

  rightservo.write(-180);

  delay(1000\*l);

  stop();

}

void gotoangle(float angle){

  int t = sqrt(sq(angle\*300\*5.7));

  if (angle > 0){

    leftservo.write(-180);

    rightservo.write(-180);

    delay(t);

  }

  else if (angle < 0){

    leftservo.write(180);

    rightservo.write(180);

    delay(t);

  }

  else {

    delay(1000);

  }

  stop();

}

void stop(){

  //stop moving

  leftservo.write(90);

  rightservo.write(90);

}

void loop() {

}