/\* fkrt el new place btt2sm l 3 hagat

1- awl haga ene a3rf el mkan el gdeed w a3mlo save

2- tany haga ene aroo7 el mkan el gdeed dh b3d lma a5las el mkan ele na feh

3- talt haga ene a3rf amshe fel mkan el gdeed btre2a mo3yana

4- lazm a3ml check na ele mkan dh ro7to abl keda wla l2

5- lam a5las mkan aroo7 lmakn tany bs a5tar ele mkan ele yb2a orayb mne

\*/

int x\_y\_1[100][2],x\_y\_2[100][2],x\_y\_3[100][2],x\_y\_4[100][2];

int x\_axis,y\_axis; //na m3rfhom abl keda fel orention

int distance1,distance2,distance3,time\_in\_y\_axis,wall\_distance;

int begins = -1; //deh le x1,x2,y1,y2

int ends ; /\* deh le x3,x4,y3,y4 \*/

int in1,in2,in3,in4;

void setup() {

// put your setup code here, to run once:

}

void loop() {

check();

turn\_left();

move\_forward\_2(); // deh function lel turn\_left wel turn\_right

turn\_left();

stop\_moving();

x\_y\_4[begins][0] = x\_axis;

x\_y\_4[begins][1] = y\_axis;

move\_forward();

check();

ends=begins+1;

x\_y\_3[ends][0] = x\_axis;

x\_y\_3[ends][1] = y\_axis;

turn\_right();

move\_forward\_2(); // deh function lel turn\_left wel turn\_right

turn\_right();

stop\_moving();

x\_y\_4[begins][0] = x\_axis;

x\_y\_4[begins][1] = y\_axis;

move\_forward();

x\_y\_3[ends][0] = x\_axis;

x\_y\_3[ends][1] = y\_axis;

}

void begin\_moving(){

begins++;

x\_y\_1[begins][0] = x\_axis;

x\_y\_1[begins][1] = y\_axis;

move\_forward();

x\_y\_2[begins][0] = x\_axis;

x\_y\_2[begins][1] = y\_axis;

}

int ultra\_sonic(){}

void turn\_left(){}

void turn\_right(){}

void stop\_moving(){}

void move\_forward(boolean left\_or\_right){

while(1){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

distance1 = ultra\_sonic();

if(distance1 <= wall\_distance){

stop\_moving();

break;

}

new\_place(left\_or\_right);.

}

//postion();

//direction\_axis();

}

void move\_forward\_2(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

delay(time\_in\_y\_axis);

}

void check(){

int place;

distance2 = ultra\_sonic();

distance3 = ultra\_sonic();

if(distance1 < wall\_distance &&( distance2 < wall\_distance || distance2 < wall\_distance)){

while(1){

stop\_moving();

place = which\_place\_to\_go();

if( (new\_places[place][1] > y\_axis - 20) && (new\_places[place][1] <= y\_axis + 20) ){

while(1){

move\_backward();

if(left\_or\_right){distance2 = ultra\_sonic(); if(distance2>wall\_distance){break;}}

else{distance3 = ultra\_sonic(); if(distance3>wall\_distance){break;}}

}

}

else{ go\_to\_place(place); }

}

}

}

void new\_place(){

if(left\_or\_right){

/\*

distance2 = ultra\_sonic();

if(distance2 > wall\_distance){

save\_x\_axis = x\_axis; //bsave el mkan l7d lma ala2e ostacle fa a3mlo save fel variable new\_places

save\_y\_axis = y\_axis; //bsave el mkan l7d lma ala2e ostacle fa a3mlo save fel variable new\_places

}

else if(distance2 < wall\_distance){

new\_places[place\_number][0] = save\_x\_axis;

new\_places[place\_number][1] = save\_y\_axis;

//where\_new\_place[place\_number] = 4 ;

place\_number++;

}

\*/

save\_new\_place(trig\_pin\_2 , echo\_pin\_2 , 0); // ana hna sh8al fe distance 2 3shan na mash fel -ve x axis

if(x\_axis < x\_y\_3[ends][0]){ save\_new\_place(trig\_pin\_3 , echo\_pin\_3 , 1);} // ana hna sh8al fe distance 3 3shan na mash fel -ve x axis bs 3det msafa mo3yana

}

else{

save\_new\_place(trig\_pin\_3 , echo\_pin\_3 , 1);

if(x\_axis > x\_y\_4[ends][0]){ save\_new\_place(trig\_pin\_2 , echo\_pin\_2 , 0);} // ana hna sh8al fe distance 2 3shan na mash fel -ve x axis bs 3det msafa mo3yana

}

}

void save\_new\_place(int trig\_pin , int echo\_pin , int where){

int distances = ultra\_sonic();

if(distances > wall\_distance){

save\_x\_axis = x\_axis; //bsave el mkan l7d lma ala2e ostacle fa a3mlo save fel variable new\_places

save\_y\_axis = y\_axis; //bsave el mkan l7d lma ala2e ostacle fa a3mlo save fel variable new\_places

}

else if(distances < wall\_distance){

new\_places[place\_number][0] = save\_x\_axis;

new\_places[place\_number][1] = save\_y\_axis;

where\_new\_place[place\_number] = where; // if where\_new\_place = 1 so i found place by distance 2 when ia move in +ve x axis or by distance 3 when i move in -ve x axis

place\_number++;

}

}

int which\_place\_to\_go(){

int no\_of\_places,x,y,straight\_line,minumum=322427667;

while(no\_of\_places<place\_number){

x = x\_axis - new\_places[no\_of\_places][0];

y = y\_axis - new\_places[no\_of\_places][0];

straight\_line = x\*x + y\*y;

if(straight\_line < minumum){

minumum = no\_of\_places;

}

no\_of\_places++;

}

return minumum;

}

void go\_to\_place(int place ){

static boolean R\_L = left\_or\_right;

while(1){

if(R\_L){

stop\_moving();

turn\_left();

R\_L = move\_forward\_with\_wall(R\_L);

}

else{

stop\_moving();

turn\_right();

R\_L = move\_forward\_with\_wall(R\_L,place);

}

}

}

boolean move\_forward\_with\_wall(boolean pos , int place){

while(1){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

if(pos){

distance1 = ultra\_sonic();

distance2 = ultra\_sonic();

if(distance1 < wall\_distance){return true}

else if(distance2 > wall\_distance){return false }

else if( ((new\_places[place][1] > y\_axis - 20) && (new\_places[place][1] <= y\_axis + 20)) && ((new\_places[place][0] > x\_axis - 20) && (new\_places[place][0] <= x\_axis + 20)) ) { enter\_the\_new\_place(); }

}

else{

distance1 = ultra\_sonic();

distance3 = ultra\_sonic();

if(distance1 < wall\_distance){return false}

else if(distance3 > wall\_distance){return true }

}

}

}

void enter\_the\_new\_place(){

if

}