1-

void setup() {

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

}

make\_right\_orention(); //e3dl nfsk

void loop() {

}

void make\_right\_orention(){

// 7alf\_shmal();

while(1){

move\_forward();

if(distance1<corner){

half\_shmal(); //90 degree

break;

}

}

distance2 = sonarsensor();

oriention = distance2;

while(1){

move\_forward();

delay(500);

distance2 = sonarsensor();

if(distance1>corner){

if(distance2<oriention){

turn\_left(,); //turn\_left bzwya m7soba

}

else if(distance2>oriention){

turn\_right(,); //turn\_right bzwya m7soba

}

else if(distance2==oriention){

break;

}

oriention = distance2;

}

else if(distance1<corner){

turn\_random();

break;

}

}

}

void move\_forward(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

}

2-

void setup() {

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

}

go\_to\_corner();

void loop() {

}

void turn\_right(){}

void turn\_left(){}

void move\_forward(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

}

void go\_to\_corner(){

while(1){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

if(distance1 <= corner){

if(distance3 <=corner ){

turn\_right(); // 90 degree

break;

}

else if( distance2 <= corner ){

turn\_left(); // 90 degree

break;

}

else{

turn\_right(); /\* aw \*/ turn\_left();

}

}

}

}

3-

int x\_axis;

int y\_axis;

int detectState;

const int encoderIn = 8;

boolean p\_x;

void setup() {

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

Serial.begin(9600);

pinMode(encoderIn, INPUT); //Set pin 8 as input

}

void loop() {

// put your main code here, to run repeatedly:

x\_axis = -3;

Serial.println(x\_axis);

delay(500);

}

void postion(){

detectState=digitalRead(encoderIn);

if (detectState == HIGH) {

}

}

void move\_forward(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

if(p\_x==true){

+ve;

}

}

void move\_backward(){

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

}

void turn\_right(){

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

angle\_x\_right= angle\_x\_right + -90;

sum\_x\_angle = angle\_x\_right + angle\_x\_left;

if(sum\_x\_angle == 0){

angle\_x\_right = 0;

angle\_x\_left = 0;

}

else if(angle\_x\_right == -180){

postion\_y = !postion\_y;

}

angle\_y\_right=angle\_y\_right-90;

sum\_y\_angle=angle\_y\_right+angle\_y\_left;

if(sum\_y\_angle == 0){

angle\_y\_right=0;

angle\_y\_left=0;

}

else if(angle\_y\_right == -180){

p\_x=!p\_x;

angle\_y\_right=0;

}

}

void turn\_left(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

angle\_x\_left= angle\_x\_left + 90;

sum\_x\_angle = angle\_x\_right + angle\_x\_left;

if(sum\_x\_angle == 0){

angle\_x\_right = 0;

angle\_x\_left = 0;

}

else if(angle\_x\_left== 180){

postion\_y = !postion\_y;

}

angle\_y\_left=angle\_y\_left+90;

sum\_y\_angle=angle\_y\_right+angle\_y\_left;

if(sum\_y\_angle == 0){

angle\_y\_right=0;

angle\_y\_left=0;

}

else if(angle\_y\_left == 180){

p\_x=!p\_x;

angle\_y\_left=0;

}

}

3.2-

int in1,in2,in3,in4;

int x\_axis;

int y\_axis,count=0;

int encoder\_x\_axis,angle\_x\_left=0,angle\_x\_right=0,sum\_x\_angle=0;

int encoder\_y\_axis,angle\_y\_left=0,angle\_y\_right=0,sum\_y\_angle=0;

const int encoderIn1 = 8;

const int encoderIn2 = 8;

boolean postion\_x = HIGH;

boolean postion\_y = HIGH;

boolean postion\_x\_or\_postion\_y = HIGH;

void setup() {

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

Serial.begin(9600);

pinMode(encoderIn1, INPUT);

pinMode(encoderIn2, INPUT);

Serial.println(".................................................");

delay(50);

}

void loop() {

move\_forward();

turn\_left();

move\_forward();

turn\_left();

move\_forward();

turn\_left();

move\_forward();

turn\_right();

move\_forward();

turn\_right();

move\_forward();

turn\_right();

move\_forward();

turn\_left();

move\_forward();

turn\_right();

move\_forward();

turn\_right();

move\_forward();

turn\_left();

move\_forward();

turn\_right();

move\_forward();

turn\_right();

move\_forward();

turn\_left();

move\_forward();

turn\_left();

move\_forward();

turn\_left();

move\_forward();

delay(5000000000000000000000);

}

void direction\_axis(){

if(postion\_x\_or\_postion\_y){

if( postion\_x ){

Serial.println("postive x-axis");

}

else{

Serial.println("negative x-axis");

}

}

else{

if( postion\_y ){

Serial.println("postive y-axis");

}

else{

Serial.println("negative y-axis");

}

}

}

void postion(){

encoder\_x\_axis=digitalRead(encoderIn1);

encoder\_y\_axis=digitalRead(encoderIn2);

if (encoder\_x\_axis == HIGH) {

if( postion\_x ){

x\_axis++;

}

else if(postion\_x){

x\_axis--;

}

}

if (encoder\_y\_axis == HIGH) {

if( postion\_y){

y\_axis++;

}

else if(postion\_y){

y\_axis--;

}

}

}

void move\_forward(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

postion();

direction\_axis();

}

void move\_backward(){

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

}

void turn\_right(){

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

angle\_x\_right= angle\_x\_right + -90;

sum\_x\_angle = angle\_x\_right + angle\_x\_left;

if(sum\_x\_angle == 0){

angle\_x\_right = 0;

angle\_x\_left = 0;

}

else if(angle\_x\_right == -180){

postion\_y = !postion\_y;

angle\_x\_right = 0;

}

angle\_y\_right=angle\_y\_right-90;

sum\_y\_angle=angle\_y\_right+angle\_y\_left;

if(sum\_y\_angle == 0){

angle\_y\_right=0;

angle\_y\_left=0;

}

else if(angle\_y\_right == -180){

postion\_x=!postion\_x;

angle\_y\_right=0;

}

postion\_x\_or\_postion\_y=!postion\_x\_or\_postion\_y;

postion();

}

void turn\_left(){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

if(count>0){

angle\_x\_left= angle\_x\_left + 90;

sum\_x\_angle = angle\_x\_right + angle\_x\_left;

if(sum\_x\_angle == 0){

angle\_x\_right = 0;

angle\_x\_left = 0;

}

else if(angle\_x\_left == 180){

postion\_y = !postion\_y;

angle\_x\_left=0;

}

}

angle\_y\_left=angle\_y\_left+90;

sum\_y\_angle=angle\_y\_right+angle\_y\_left;

if(sum\_y\_angle == 0){

angle\_y\_right=0;

angle\_y\_left=0;

}

else if(angle\_y\_left == 180){

postion\_x=!postion\_x;

angle\_y\_left=0;

}

postion\_x\_or\_postion\_y=!postion\_x\_or\_postion\_y;

count++;

postion();

}

4- moving\_\_with\_no\_ostacles\_

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(){

while(1){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

distance1 = ultra\_sonic();

if(distance1 <= wall\_distance){

stop\_moving();

break;

}

}

//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(){

distance2 = ultra\_sonic();

distance3 = ultra\_sonic();

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

while(1){

stop\_moving();

}

}

}

5- moving with one obstacle

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 ostacle\_x; //dh vaiable be save el makan ele abl ele ostacle;

int ostacle\_y; //dh vaiable be save el makan ele abl ele ostacle;

int in1,in2,in3,in4;

boolean finding\_ostacle = true,left\_or\_right;

int time\_to\_end\_obstacle;

void setup() {

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

}

void loop() {

check();

turn\_left();

move\_forward\_2(time\_in\_y\_axis); // 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(true);

check();

ends=begins+1;

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

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

turn\_right();

move\_forward\_2(time\_in\_y\_axis); // 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(false);

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(false);

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;

}

obstacle(left\_or\_right);

}

//postion();

//direction\_axis();

}

void move\_forward\_2(int time\_y){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

delay(time\_y);

}

void check(){

distance2 = ultra\_sonic();

distance3 = ultra\_sonic();

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

while(1){

stop\_moving();

}

}

}

void obstacle(boolean left\_or\_right){

if(finding\_ostacle){

if(left\_or\_right){

distance2 = ultra\_sonic();

if(distance2 <= wall\_distance){

ostacle\_x = x\_axis;

ostacle\_y = y\_axis;

finding\_ostacle = false;

}

}

else{

distance3 = ultra\_sonic();

ostacle\_x = x\_axis;

ostacle\_y = y\_axis;

finding\_ostacle = false;

}

}

else{

if(x\_axis > ostacle\_x ){

if(left\_or\_right == false){

turn\_right();

move\_forward\_2(time\_to\_end\_obstacle);

turn\_left();

move\_forward(false); // na tla3 l2odam +ve x axis

}

else{

turn\_left();

move\_forward\_2(time\_to\_end\_obstacle);

turn\_right();

move\_forward(true); // na rag3 lwara -ve x axis

}

finding\_ostacle = true;

}

}

}

6- moving\_\_with\_obstacle\_at\_the\_begining\_or\_the\_end\_

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 ostacle\_x; //dh vaiable be save el makan ele abl ele ostacle;

int ostacle\_y; //dh vaiable be save el makan ele abl ele ostacle;

int in1,in2,in3,in4;

boolean finding\_ostacle = true,left\_or\_right;

int time\_to\_end\_obstacle;

void setup() {

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

}

void loop() {

check(false);

turn\_left();

move\_forward\_2(time\_in\_y\_axis); // 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(true);

check(true);

ends=begins+1;

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

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

turn\_right();

move\_forward\_2(time\_in\_y\_axis); // 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(false);

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(false);

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\_backward(){}

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;

}

obstacle(left\_or\_right);

}

//postion();

//direction\_axis();

}

void move\_forward\_2(int time\_y){

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

delay(time\_y);

}

void check(boolean check\_left\_or\_right){

distance1 = ultra\_sonic();

if(distance1 < wall\_distance)

{

if(check\_left\_or\_right == false){

distance3 = ultra\_sonic();

if(distance3 <= wall\_distance){

end\_moving(check\_left\_or\_right);

}

}

else{

distance2 = ultra\_sonic();

if(distance2 <= wall\_distance){

end\_moving(check\_left\_or\_right);

}

}

}

}

void obstacle(boolean left\_or\_right){

if(finding\_ostacle){

if(left\_or\_right){

distance2 = ultra\_sonic();

if(distance2 <= wall\_distance){

ostacle\_x = x\_axis;

ostacle\_y = y\_axis;

finding\_ostacle = false;

}

}

else{

distance3 = ultra\_sonic();

ostacle\_x = x\_axis;

ostacle\_y = y\_axis;

finding\_ostacle = false;

}

}

else{

if(x\_axis > ostacle\_x ){

if(left\_or\_right == false){

turn\_right();

move\_forward\_2(time\_to\_end\_obstacle);

turn\_left();

move\_forward(false); // na tla3 l2odam +ve x axis

}

else{

turn\_left();

move\_forward\_2(time\_to\_end\_obstacle);

turn\_right();

move\_forward(true); // na rag3 lwara -ve x axis

}

finding\_ostacle = true;

}

}

}

void end\_moving(boolean check\_left\_or\_right){

if(check\_left\_or\_right){

while(1){

move\_backward(); // lsa m3mlthash

distance2 = ultra\_sonic();

if(distance2 > wall\_distance){

break;

}

else if(x\_axis > x\_y\_1[0][0]){

while(1){

stop\_moving();

}

}

}

}

else{

while(1){

move\_backward();

distance3 = ultra\_sonic();

if(distance3 > wall\_distance){

break;

}

else if(x\_axis > x\_y\_1[0][0]){

while(1){

stop\_moving();

}

}

}

}

}

7-

/\* 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;

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);

}

}

}

boolean move\_forward\_with\_wall(boolean pos){

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()

}

else{

distance1 = ultra\_sonic();

distance3 = ultra\_sonic();

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

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

}

}

}

7- new place with no obstacle :

/\* 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(){

begin\_function:

begins++;

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

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

move\_forward\_begin();

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\_begin(){

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(true);.

new\_place(false);.

}

}

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){

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) && ( straight\_line !=0) ){

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(){

// el mkan fel ymeen bel nsba l distance 2

if(where\_new\_place[place\_number] == 1 ){

old\_y\_axis = y\_axis;

old\_x\_axis = x\_axis;

move\_forward\_2(); // hmshee forward haga bseta awe

new\_y\_axis = y\_axis;

new\_x\_axis = x\_axis

if( old\_y\_axis == new\_y\_axis ) {

if( old\_x\_axis < new\_x\_axis ){move\_forward\_2(); turn\_right(); move\_forward\_2(); turn\_right();} // awl w tany move forward na bmshe l7 lma ml2eesh ostacle 3shan alf

else{move\_forward\_2(); turn\_left(); move\_forward\_2(); turn\_right(); } // awl w tany move forward na bmshe l7 lma ml2eesh ostacle 3shan alf

}

else if( old\_y\_axis < new\_y\_axis ){ move\_backward(); turn\_left();} // lma hmshe backward na hmsh l7d lma ml2eesh ostacle y3ny lazm a5le el function keda

else {turn\_right();}

}

// el mkan fel shmal bel nsba l distance 3

else if(where\_new\_place[place\_number] == 0 ){

old\_y\_axis = y\_axis;

old\_x\_axis = x\_axis;

move\_forward\_2(); // hmshee forward haga bseta awe

new\_y\_axis = y\_axis;

new\_x\_axis = x\_axis

if( old\_y\_axis == new\_y\_axis ) {

if( old\_x\_axis < new\_x\_axis ){move\_forward\_2(); turn\_left() move\_forward\_2(); turn\_right();} // awl w tany move forward na bmshe l7 lma ml2eesh ostacle 3shan alf

else{move\_forward\_2(); turn\_right(); move\_forward\_2(); turn\_right(); } // awl w tany move forward na bmshe l7 lma ml2eesh ostacle 3shan alf

}

else if( old\_y\_axis > new\_y\_axis ){ move\_backward(); turn\_left();} // lma hmshe backward na hmsh l7d lma ml2eesh ostacle y3ny lazm a5le el function keda

else {turn\_right(); }

}

new\_places[place\_number][0] = 0;

new\_places[place\_number][0] = 0;

goto begin\_function;

}

8-