#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, TouchSensor, sensorEV3\_Ultrasonic)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, driveRight, encoder)

#pragma config(Motor, motorC, crane, tmotorEV3\_Medium, PIDControl, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor =

int speedA = 100

int speedB = 0

int defaultSpeed = 50

int distanceDetect = 15

//START

task begin(){

//MOVE TO WHITE LINE

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect white line - > rotate 180

if(SensorValue[ColorSensor] >= whiteColor){

setMotorSpeed(motorA, speedA);

setMotorSpeed(motorB, speedB);

//moveMotorTarget(motorA,180,defaultSpeed);

}

return;

}

//ROTATE

task rotate(){

setMotorSpeed(motorA, speedA);

setMotorSpeed(motorB, speedB);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//ATTACK

task attack(){

/\*swing axe, fork lift, nothing?\*/

}

task main(){

StartTask(begin);

while(1=1){

//Drive forward

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect Enemy or white line

if(getUSDistance(TouchSensor) <= distanceDetect){

StartTask(attack);

}else if(SensorValue[ColorSensor] >= whiteColor){

StartTask(rotate);

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, TouchSensor, sensorEV3\_Ultrasonic)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, driveRight, encoder)

#pragma config(Motor, motorC, crane, tmotorEV3\_Medium, PIDControl, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor =

int speedA = 100

int speedB = 0

int defaultSpeed = 50

int distanceDetect = 15

int turningSpeedA = 55

int turningSpeedB = -25

//START

task begin(){

//MOVE TO WHITE LINE

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect white line - > rotate 180

if(SensorValue[ColorSensor] >= whiteColor){

setMotorSpeed(motorA, speedA);

setMotorSpeed(motorB, speedB);

//moveMotorTarget(motorA,180,defaultSpeed);

}

return;

}

//ROTATE

task rotate(){

setMotorSpeed(motorA, speedA);

setMotorSpeed(motorB, speedB);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//ATTACK

task attack(){

/\*(should be stationary) swing axe, fork lift, push out of way with long rod thats time based, spin, avoid?\*/

}

task main(){

StartTask(begin);

while(1=1){

//Drive forward

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or turn

if(getUSDistance(TouchSensor) <= distanceDetect){

StartTask(attack);

}else{

setMotorSpeed(motorA, turningSpeedA);

setMotorSpeed(motorB, turningSpeedB);

//Wait1Msec(500);

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

if((SensorValue[ColorSensor] >= whiteColor)){

StartTask(rotate);

}

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, TouchSensor, sensorEV3\_Ultrasonic)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, driveRight, encoder)

#pragma config(Motor, motorC, crane, tmotorEV3\_Medium, PIDControl, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor = 20;

int speedARotate = 100;

int speedBRotate = 0;

int defaultSpeed = 50;

int distanceDetect = 15;

int turningSpeedA = 55;

int turningSpeedB = -25;

//ROTATE

task rotate(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//START

task begin(){

//MOVE TO WHITE LINE

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect white line - > rotate 180

if(SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

}

return;

}

//ATTACK

task attack(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

wait1Msec(3000);

return;

}

task main(){

startTask(begin);

while(1==1){

//Drive forward

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or turn

if(getUSDistance(TouchSensor) <= distanceDetect){

startTask(attack);

}else if (SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

setMotorSpeed(motorA,turningSpeedA);

setMotorSpeed(motorB,turningSpeedB);

//wait1Msec(200);

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, TouchSensor, sensorEV3\_Ultrasonic)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, driveRight, encoder)

#pragma config(Motor, motorC, crane, tmotorEV3\_Medium, PIDControl, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor = 20;

int speedARotate = 100;

int speedBRotate = 0;

int defaultSpeed = 50;

int distanceDetect = 15;

int turningSpeedA = 55;

int turningSpeedB = -25;

//ROTATE

task rotate(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//START

task begin(){

//MOVE TO WHITE LINE

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect white line - > rotate 180

if(SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

}

return;

}

//ATTACK

task attack(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

wait1Msec(3000);

return;

}

//TURN

task attack(){

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//wait1Msec(100);

setMotorSpeed(motorA,turningSpeedA);

setMotorSpeed(motorB,turningSpeedB);

return;

}

task main(){

startTask(begin);

while(1==1){

//Turn then drive forward

startTask(turn);

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or turn

if(getUSDistance(TouchSensor) <= distanceDetect){

startTask(attack);

}else if (SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, TouchSensor, sensorEV3\_Ultrasonic)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, driveRight, encoder)

#pragma config(Motor, motorC, crane, tmotorEV3\_Medium, PIDControl, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor = 20;

int speedARotate = 100;

int speedBRotate = 0;

int defaultSpeed = 50;

int distanceDetect = 15;

int turningSpeedA = 55;

int turningSpeedB = -25;

int increasedSpeed = 200;

//ROTATE

task rotate(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//START

task begin(){

//MOVE TO WHITE LINE

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect white line - > rotate 180

if(SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

}

return;

}

//ATTACK SIDE

task attackSide(){

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

wait1Msec(3000);

return;

}

//ATTACK FRONT

task attackFront(){

setMotorSpeed(motorA, increasedSpeed);

setMotorSpeed(motorB, increasedSpeed);

return;

}

//TURN

task attack(){

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//wait1Msec(100);

setMotorSpeed(motorA,turningSpeedA);

setMotorSpeed(motorB,turningSpeedB);

return;

}

task main(){

startTask(begin);

while(1==1){

//Turn then drive forward

startTask(turn);

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or rotate

if(getUSDistance(SensorValue[ColorSensor] >= whiteColor){

startTask(rotate);

}else if (getUSDistance(TouchSensor) <= distanceDetect){

startTask(attackFront);

}else if (getTouchValue(nDeviceIndex)=1){

startTask(attackSide);

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, DistanceSensor, sensorEV3\_Ultrasonic)

#pragma config(Sensor, S3, LeftTouchSensor, sensorEV3\_Touch)

#pragma config(Sensor, S4, RightTouchSensor, sensorEV3\_Touch)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, reversed, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, reversed, driveRight, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor = 25;

int speedARotate = 50;

int speedBRotate = -25;

int defaultSpeed = 20;

int distanceDetect = 15;

int increasedSpeed = 200;

int spinSpeedA = 100

int spinSpeedB = -45

//ROTATE

task rotate(){

setMotorSpeed(motorA, -defaultSpeed);

setMotorSpeed(motorB, -defaultSpeed);

//wait1Msec();

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//ATTACK SIDE

task attackSide(){

setMotorSpeed(motorA, spinSpeedA)

setMotorSpeed(motorB, spinSpeedB);

return;

}

//ATTACK FRONT

task attackFront(){

setMotorSpeed(motorA, increasedSpeed);

setMotorSpeed(motorB, increasedSpeed);

return;

}

task main(){

while(1==1){

displayCenteredBigTextLine(1, " Color Value: %d",SensorValue[ColorSensor]);

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or rotate

if(SensorValue[S1] >= whiteColor){

startTask(rotate);

displayBigTextLine(4, "Cum detected");

}else if (getUSDistance(DistanceSensor) <= distanceDetect){

startTask(attackFront);

displayBigTextLine(4, "Charging");

}else if (getBumpedValue(RightTouchSensor)==1 || getBumpedValue(LeftTouchSensor) == 1){

while(getUSDistance(DistanceSensor)>distanceDetect){

startTask(attackSide);

displayBigTextLine(4, "Touch sensor triggered");

}

}

}

}

#pragma config(Sensor, S1, ColorSensor, sensorEV3\_Color)

#pragma config(Sensor, S2, DistanceSensor, sensorEV3\_Ultrasonic)

#pragma config(Sensor, S3, LeftTouchSensor, sensorEV3\_Touch)

#pragma config(Sensor, S4, RightTouchSensor, sensorEV3\_Touch)

#pragma config(Motor, motorA, LeftWheel, tmotorEV3\_Large, PIDControl, reversed, driveLeft, encoder)

#pragma config(Motor, motorB, RightWheel, tmotorEV3\_Large, PIDControl, reversed, driveRight, encoder)

//\*!!Code automatically generated by 'ROBOTC' configuration wizard !!\*//

//!!Code automatically generated by 'ROBOTC' configuration wizard !!

/\* TO DO:

\*/

int whiteColor = 25;

int speedARotate = 50;

int speedBRotate = -25;

int defaultSpeed = 20;

int distanceDetect = 15;

int increasedSpeed = 200;

int spinSpeedA = 100

int spinSpeedB = -45

//ROTATE

task rotate(){

setMotorSpeed(motorA, -defaultSpeed);

setMotorSpeed(motorB, -defaultSpeed);

//wait1Msec();

setMotorSpeed(motorA, speedARotate);

setMotorSpeed(motorB, speedBRotate);

//moveMotorTarget(motorA,180,defaultSpeed);

return;

}

//ATTACK SIDE

task attackSide(){

setMotorSpeed(motorA, spinSpeedA)

setMotorSpeed(motorB, spinSpeedB);

return;

}

//ATTACK FRONT

task attackFront(){

setMotorSpeed(motorA, increasedSpeed);

setMotorSpeed(motorB, increasedSpeed);

return;

}

task main(){

while(1==1){

displayCenteredBigTextLine(1, " Color Value: %d",SensorValue[ColorSensor]);

setMotorSpeed(motorA,defaultSpeed);

setMotorSpeed(motorB,defaultSpeed);

//Detect enemy or rotate

if(SensorValue[S1] >= whiteColor){

startTask(rotate);

displayBigTextLine(4, "Cum detected");

}else if (getUSDistance(DistanceSensor) <= distanceDetect){

startTask(attackFront);

displayBigTextLine(4, "Charging");

}else if (getBumpedValue(RightTouchSensor)==1 || getBumpedValue(LeftTouchSensor) == 1){

while(getUSDistance(DistanceSensor)>distanceDetect){

startTask(attackSide);

displayBigTextLine(4, "Touch sensor triggered");

}

}else if (SensorValue[S1] >= outside arena color){

setMotorSpeed(motorA, -200)

setMotorSpeed(motorB, -200);

displayBigTextLine(4, "Outside");

wait1Msec(1500);

while(getUSDistance(DistanceSensor)>distanceDetect){

startTask(attackSide);

}

}

}

}