Nombre del estudiante: RB20

Asignación:

notas:

Nombre del Projecto: RB20

Tipo de proyecto: C++

Date: Mon Jun 10 2024

```
#pragma region VEXcode Generated Robot Configuration
// Make sure all required headers are included.
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <math.h>
#include <string.h>


#include "vex.h"

using namespace vex;

// Brain should be defined by default
brain Brain;


// START IQ MACROS
#define waitUntil(condition) \
do {                                                                       \
wait(5, msec); \
  }while (!(condition))

#define repeat(iterations) \
for (int iterator = 0; iterator < iterations; iterator++)
// END IQ MACROS


// Robot configuration code.
inertial BrainInertial = inertial();
distance SI = distance(PORT5);
distance SC = distance(PORT3);
distance SD = distance(PORT11);
motor LeftDriveSmart = motor(PORT1, 1, false);
motor RightDriveSmart = motor(PORT12, 1, true);
drivetrain Drivetrain = drivetrain(LeftDriveSmart, RightDriveSmart, 200, 173, 76, mm, 1);
motor VOLANTE = motor(PORT6, true);
colorsensor COLOR = colorsensor(PORT4);

#pragma endregion VEXcode Generated Robot Configuration

// Include the IQ Library
#include "vex.h"

// Allows for easier use of the VEX Library
using namespace vex;

float myVariable, giro, grados;

int mathRandomInt(float a, float b) {
if (a > b) {
// Swap a and b to ensure a is smaller.
float c = a;
a = b;
b = c;
  }
int tmpA = static_cast<int>(a);
int tmpB = static_cast<int>(b);
int r = tmpA + rand() / (RAND_MAX / (tmpB - tmpA + 1));
return r;
}

// "when started" hat block
int whenStarted1() {
while (true) {
Drivetrain.drive(forward);
Drivetrain.setDriveVelocity(100.0, percent);
if (SC.objectDistance(mm) < 280.0) {
giro = static_cast<float>(mathRandomInt(1.0, 1.0));
grados = static_cast<float>(mathRandomInt(25.0, 25.0));
if (giro == 1.0) {
VOLANTE.setVelocity(100.0, percent);
VOLANTE.spinFor(reverse, grados, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(forward, 40.0, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(reverse, grados, degrees, true);
      }
    }
if (SI.objectDistance(mm) < 100.0) {
giro = static_cast<float>(mathRandomInt(1.0, 2.0));
grados = static_cast<float>(mathRandomInt(25.0, 25.0));
if (giro == 1.0) {
VOLANTE.setVelocity(100.0, percent);
VOLANTE.spinFor(reverse, grados, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(forward, 40.0, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(reverse, grados, degrees, true);
      }
    }
if (SD.objectDistance(mm) < 100.0) {
giro = static_cast<float>(mathRandomInt(1.0, 1.0));
grados = static_cast<float>(mathRandomInt(25.0, 25.0));
if (giro == 1.0) {
VOLANTE.setVelocity(100.0, percent);
VOLANTE.spinFor(forward, grados, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(reverse, 40.0, degrees, true);
wait(0.5, seconds);
VOLANTE.spinFor(forward, grados, degrees, true);
      }
    }
wait(20, msec);
  }
```

```
return 0;
}


int main() {
// initialize the random number system
srand(Brain.Timer.system());

whenStarted1();
}
```