#pragma config(Motor, port1, rightDrive, tmotorVex393\_HBridge, openLoop, driveRight)

#pragma config(Motor, port3, lowhangmotor, tmotorVex393\_MC29, openLoop)

#pragma config(Motor, port4, newmotor, tmotorVex393\_MC29, openLoop, reversed)

#pragma config(Motor, port5, sidedrive, tmotorVex393\_MC29, openLoop)

#pragma config(Motor, port6, ClawMotor, tmotorVex393\_MC29, openLoop)

#pragma config(Motor, port7, armmotor, tmotorVex393\_MC29, openLoop)

#pragma config(Motor, port8, highhangmotor, tmotorVex393\_MC29, openLoop)

#pragma config(Motor, port10, leftDrive, tmotorVex393\_HBridge, openLoop, reversed, driveLeft)

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

// This code is for the VEX cortex platform

#pragma platform(VEX2)

// Select Download method as "competition"

#pragma competitionControl(Competition)

//Main competition background code...do not modify!

#include "Vex\_Competition\_Includes.c"

void driveTrain();

void armControl();

void clawControl();

void pre\_auton()

{

 bStopTasksBetweenModes = true;

}

task usercontrol()

{

 while (true)

 {

 sleep(20);

 driveTrain();

 armControl();

 clawControl();

 }

}

void driveTrain()

{

 setMotor(rightDrive, vexRT[Ch3]);

 setMotor(leftDrive, vexRT[Ch2]);

}

void armControl()

{

 if(vexRT[Btn6U])

 {

 setMotor(armmotor, 64);

 setMotor(newmotor, 64);

 }

 else if(vexRT[Btn6D])

 {

 setMotor(armmotor, -64);

 setMotor(newmotor, -64);

 }

 else

 {

 setMotor(armmotor, 0);

 setMotor(newmotor, 0);

 }

 if(vexRT[Btn8L])

 {

 setMotor(sidedrive, 64);//drive left

 }

 else if(vexRT[Btn8R])

 {

 setMotor(sidedrive, -64);//drive right

 }

 else

 {

 setMotor(sidedrive, 0);

 }

 {

 if(vexRT[Btn7D])

 {

 setMotor(lowhangmotor, 32);// down

 }

 else if(vexRT[Btn7U])

 {

 setMotor(lowhangmotor, -32);//up

 }

 else

 {

 setMotor(lowhangmotor, 0);

 }

 }

 {

 if(vexRT[Btn8U])

 {

 setMotor(highhangmotor, -64);

 }

 else if(vexRT[Btn8D])

 {

 setMotor(highhangmotor, 32);

 }

 else

 {

 setMotor(highhangmotor, 0);

 }

 }

}

void clawControl()

{

 if(vexRT[Btn5U])

 {

 setMotor(ClawMotor, 64);

 }

 else if(vexRT[Btn5D])

 {

 setMotor(ClawMotor, -64);

 }

 else

 {

 setMotor(ClawMotor, 0);

 }

}

task autonomous()//0.5+0.5+0.5+3+1.5=6

{

 setMotor(leftDrive, 127);

 setMotor(rightDrive, 127);

 delay(500);

 setMotor(leftDrive, 0);

 setMotor(rightDrive, 0);

 setMotor(ClawMotor, 48);

 delay(500);

 setMotor(ClawMotor, 0);

 setMotor(leftDrive, -127);

 setMotor(rightDrive, -127);

 delay(500);

 setMotor(leftDrive, 100);

 setMotor(rightDrive, 100);//forward

 delay(3500);

 setMotor(leftDrive, -100);

 setMotor(rightDrive, -100);//back

 delay(500);

 setMotor(leftDrive, 64);

 setMotor(rightDrive, 64);

 delay(1000);

 setMotor(leftDrive, -100);

 setMotor(rightDrive, -100);

 delay(500);

 setMotor(leftDrive, 0);

 setMotor(rightDrive, 0);

 delay(8000);

}