#include "robot-config.h"

vex::brain brain;

vex::motor baseFLeft (vex::PORT1,vex::gearSetting::ratio18\_1,false);

vex::motor baseFRight (vex::PORT2,vex::gearSetting::ratio18\_1,false);

vex::motor flywheel (vex::PORT3,vex::gearSetting::ratio18\_1,false);

vex::motor flipper (vex::PORT4,vex::gearSetting::ratio18\_1,false);

vex::motor intake (vex::PORT5,vex::gearSetting::ratio18\_1,true);

vex::motor baseBLeft (vex::PORT6,vex::gearSetting::ratio18\_1,false);

vex::motor baseBRight (vex::PORT7,vex::gearSetting::ratio18\_1,true);

/\*---------------------------------------------------------------------------\*/

/\* \*/

/\* Description: Competition template for VCS VEX V5 \*/

/\* \*/

/\*---------------------------------------------------------------------------\*/

//Creates a competition object that allows access to Competition methods.

vex::competition;

/\*---------------------------------------------------------------------------\*/

/\* Pre-Autonomous Functions \*/

/\* \*/

/\* You may want to perform some actions before the competition starts. \*/

/\* Do them in the following function. You must return from this function \*/

/\* or the autonomous and usercontrol tasks will not be started. This \*/

/\* function is only called once after the cortex has been powered on and \*/

/\* not every time that the robot is disabled. \*/

/\*---------------------------------------------------------------------------\*/

void pre\_auton( void ) {

// All activities that occur before the competition starts

// Example: clearing encoders, setting servo positions, ...

}

/\*---------------------------------------------------------------------------\*/

/\* \*/

/\* Autonomous Task \*/

/\* \*/

/\* This task is used to control your robot during the autonomous phase of \*/

/\* a VEX Competition. \*/

/\* \*/

/\* You must modify the code to add your own robot specific commands here. \*/

/\*---------------------------------------------------------------------------\*/

void autonomous( void ) {

// ..........................................................................

// Insert autonomous user code here.

// ..........................................................................

BaseFRight.startRotateFor(590,rotationUnits::deg,70,velocityUnits::pct);

BaseBRight.rotateFor(590,rotationUnits::deg,70,velocityUnits::pct,false);

BaseFLeft.rotateFor(590,rotationUnits::deg,70,velocityUnits::pct,false);

BaseBLeft.rotateFor(590,rotationUnits::deg,70,velocityUnits::pct);

BaseFRight.startRotateFor(-190,rotationUnits::deg,40,velocityUnits::pct);

BaseBRight.rotateFor(-190,rotationUnits::deg,40,velocityUnits::pct,false);

BaseFLeft.rotateFor(190,rotationUnits::deg,40,velocityUnits::pct,false);

BaseBLeft.rotateFor(190,rotationUnits::deg,40,velocityUnits::pct);

BaseFRight.startRotateFor(1200,rotationUnits::deg,90,velocityUnits::pct);

BaseBRight.rotateFor(1200,rotationUnits::deg,90,velocityUnits::pct,false);

BaseFLeft.rotateFor(1200,rotationUnits::deg,90,velocityUnits::pct,false);

BaseBLeft.rotateFor(1200,rotationUnits::deg,90,velocityUnits::pct);

}

void usercontrol( void ) {

// User control code here, inside the loop

while (1){

BaseFLeft.spin(directionType::fwd,(Controller1.Axis3.value()),velocityUnits::pct);

BaseBLeft.spin(directionType::fwd,(Controller1.Axis3.value()),velocityUnits::pct);

BaseFRight.spin(directionType::fwd,(Controller1.Axis2.value()),velocityUnits::pct);

BaseBRight.spin(directionType::fwd,(Controller1.Axis2.value()),velocityUnits::pct);

if (Controller1.ButtonDown.pressing()){

Flywheel.spin(directionType::fwd,90,velocityUnits::pct);

}

else if (Controller1.ButtonB.pressing()){

Flywheel.stop(brakeType::coast);

}

else if (Controller1.ButtonL2.pressing()){

Intake.spin(directionType::fwd,-75,velocityUnits::pct);

}

else if (Controller1.ButtonL1.pressing()){

//Intake.stop(brakeType::coast);

Intake.spin(directionType::fwd,75,velocityUnits::pct);

}

else if (Controller1.ButtonR1.pressing()){

Flipper.spin(directionType::fwd,38,velocityUnits::pct);

}

else if (Controller1.ButtonR2.pressing()){

Flipper.stop(brakeType::coast);

}

else{

Intake.stop(brakeType::hold);

}

task::sleep(20);//Sleep the task for a short amount of time to prevent wasted resources.

}

// This is the main execution loop for the user control program.

// Each time through the loop your program should update motor + servo

// values based on feedback from the joysticks.

// ........................................................................

// Insert user code here. This is where you use the joystick values to

// update your motors, etc.

// ........................................................................

vex::task::sleep(20); //Sleep the task for a short amount of time to prevent wasted resources.

}

//

// Main will set up the competition functions and callbacks.

//

int main() {

//Run the pre-autonomous function.

pre\_auton();

//Set up callbacks for autonomous and driver control periods.

Competition.autonomous( autonomous );

Competition.drivercontrol( usercontrol );

//Prevent main from exiting with an infinite loop.

}