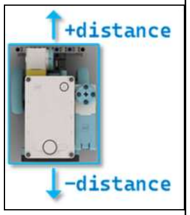
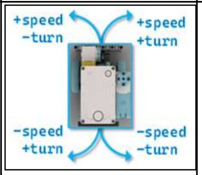
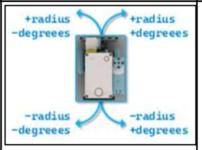
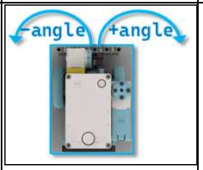
Robot Movement

Action		How Much?	Python Code	Parameters
Drive Straight		Distance	r.robot.straight(distance)	**distance:** Distance to travel (mm), robot runs at default speed example - r.robot.straight(100) - Goes straight forward 100 mm

Drive Straight or Curved	 +speed -turn
	+speed +turn
	-speed +turn
	-speed -turn
		Time	r.robot.drive(speed,turn_rate)
			wait(time)
			r.robot.stop()
				speed: Speed of the robot (mm/s)
				turn_rate: Turn rate of the robot (deg/s) 0 = Straight
				time: How long to drive (mSec)
				Drives the robot at set speed either straight or while turning
				r.robot.drive(200, 20) - Drive robot slowly forward while turning gently to right

Curve	 +radius -degreees +radius +degreees -radius -degreees -radius +degreees	Distance	r.robot.arc(radius, degrees)	**radius:** Radius of the circle (mm) **distance:** Distance to drive along the circle (mm) Drives center of wheels over specified radius circle for degrees of arc r.robot.arc(50, 180) - drives robot halfway around tight circle
Tank Turn		Angle	r.robot.turn(angle)	**angle:** Angle of the turn (deg) r.robot.turn(-90) - Make robot spin 90 degrees to the left

Attachment Motors

Side		How Much?	Python Code	Parameters
Left		Time	r.lam.run_time(speed,time)	**speed:** Speed of the motor (deg/s) **time:** Duration of motor running (mSec) r.lam.run_time(1000, 500) - runs lam (left attachment motor) clockwise fast for 0.5 seconds
Left		Angle	r.lam.run_angle(speed,angle)	**speed:** Speed of the motor (deg/s) **angle:** Angle by which the motor should rotate (deg) r.ram.run_angle(500, 45) - runs ram (right attachment motor) clockwise at half speed for 45 degrees
Left		Target Angle	r.lam.run_target(speed,target_angle)	**speed:** Speed of the motor (deg/s) **target_angle:** Angle in degrees that the motor should rotate to, angle is zero when marks on motor hub and body are aligned. Motor angles can be reset by program, see "Motors with rotation sensors" r.lam.run_target(200, -60) - runs lam to 60 degrees counter-clockwise from reference mark on
Left		Stall	r.lam.run_until_stalled(speed,duty_limit=stall_power)	**speed:** Speed of the motor (deg/s) **stall_power:** What percent power to is considered a stall (0 - 100) r.ram.run_until_stalled(1000, duty_limit=100) - runs ram clockwise until it pushes hard on something
Right		---	**For right attachment motor replace "lam" above with "ram"**	
		Maximums	**Medium motors = 1100, Large motors = 1050**	**Direction - Normally positive is clockwise rotation unless switched in robot.py near line 147**

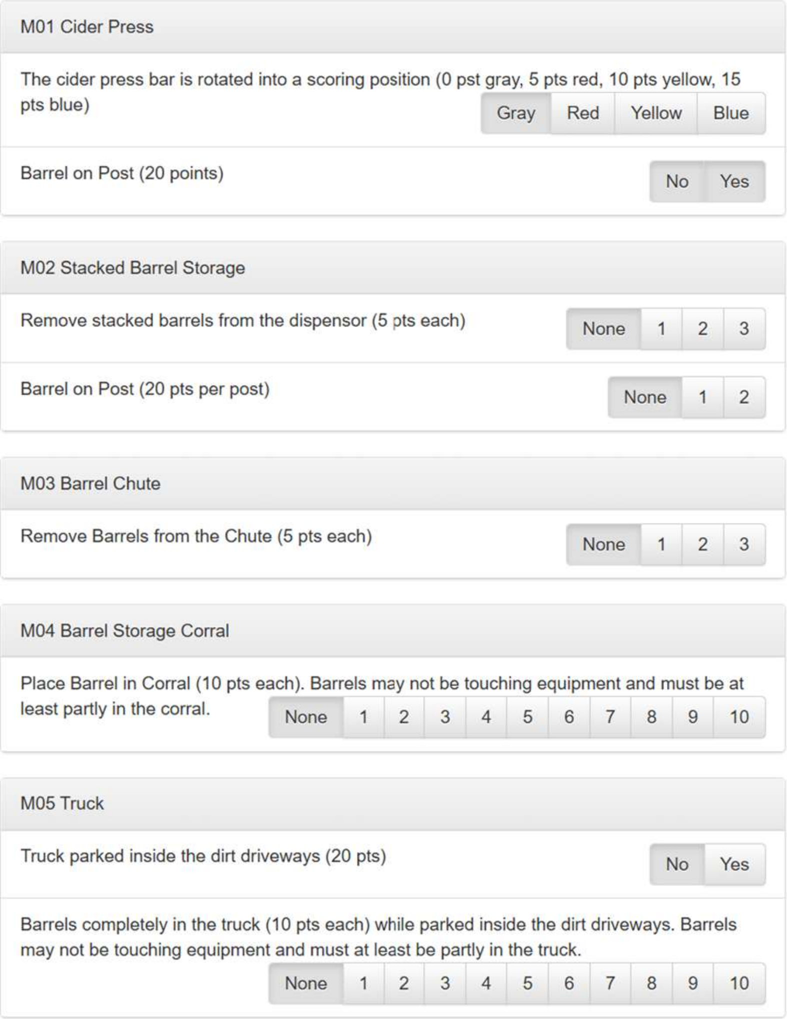
Other Commands

Action			Python Code	Parameters
Pause the Program			wait(time)	**time:** How long to pause the program (mSec) wait(500) - keeps program from going to next instruction (waiting) for 500 milliseconds = 0.5 seconds
Print to console			print("your_message")	**your_message:** Text to be displayed in the console window print("Hello World") - prints message Hello World on console display of computer Bluetoothed to bot
Color Sensor Reflectivity			value = r.rcs.reflection()	**Value:** is set to reflectivity color sensor sees, use "lcs" for left sensor Code on left followed by print(value) will color sensor reflection values from 0 to 99 on console

Setting Tire Diameter and Axle Track

To set values for your robot first measure TIRE_DIAMETER in millimeters. Second find your AXLE_TRACK by measuring space between the very center of the wheels in millimeters. Now enter these values in robot.py, you'll find these constants around program line100. A Spike bot will be about TIRE_DIAMETER = 56 and AXLE_TRACK = 82, yours may be different if you use different wheels or spacing of the motors on your bot. The exact procedure for setting your robot's values is found in pybricks help menu, search on "robotics and drive bases", then look about half way down the section for the heading "Measuring and validating the robot dimensions". If you follow this procedure your robot will drive as far and turn as much as you tell it.

https://fssfll.github.io/fssfll/spike/spring_2024/robot_game/



M01 Cider Press

The cider press bar is rotated into a scoring position (0 pst gray, 5 pts red, 10 pts yellow, 15 pts blue)

Gray | Red | Yellow | Blue

Barrel on Post (20 points)

No | Yes

M02 Stacked Barrel Storage

Remove stacked barrels from the dispensor (5 pts each)

None | 1 | 2 | 3

Barrel on Post (20 pts per post)

None | 1 | 2

M03 Barrel Chute

Remove Barrels from the Chute (5 pts each)

None | 1 | 2 | 3

M04 Barrel Storage Corral

Place Barrel in Corral (10 pts each). Barrels may not be touching equipment and must be at least partly in the corral.

None | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10

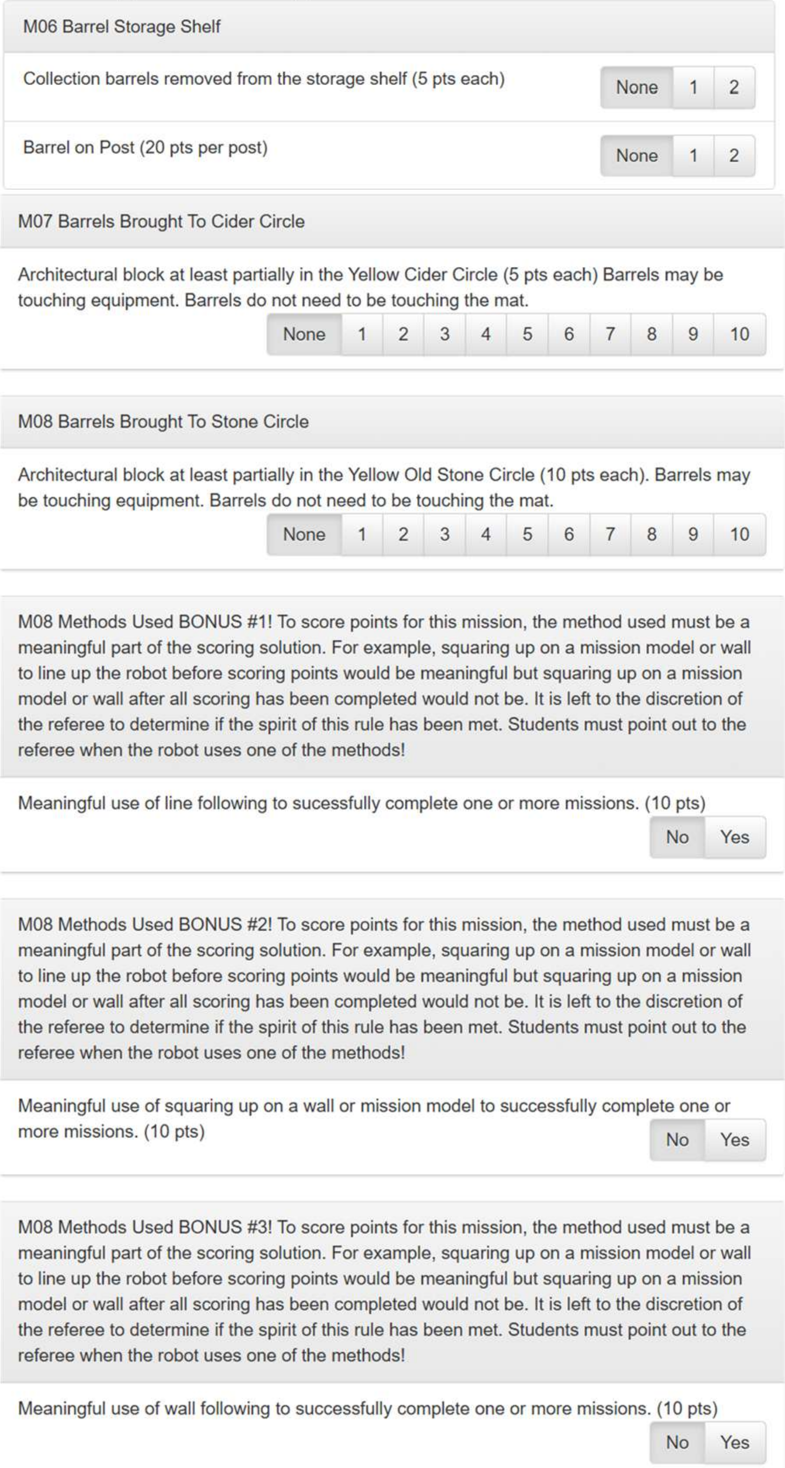
M05 Truck

Truck parked inside the dirt driveways (20 pts)

No | Yes

Barrels completely in the truck (10 pts each) while parked inside the dirt driveways. Barrels may not be touching equipment and must at least be partly in the truck.

None | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10



M06 Barrel Storage Shelf

Collection barrels removed from the storage shelf (5 pts each)

None | 1 | 2

Barrel on Post (20 pts per post)

None | 1 | 2

M07 Barrels Brought To Cider Circle

Architectural block at least partially in the Yellow Cider Circle (5 pts each) Barrels may be touching equipment. Barrels do not need to be touching the mat.

None | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10

M08 Barrels Brought To Stone Circle

Architectural block at least partially in the Yellow Old Stone Circle (10 pts each). Barrels may be touching equipment. Barrels do not need to be touching the mat.

None | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10

M08 Methods Used BONUS #1! To score points for this mission, the method used must be a meaningful part of the scoring solution. For example, squaring up on a mission model or wall to line up the robot before scoring points would be meaningful but squaring up on a mission model or wall after all scoring has been completed would not be. It is left to the discretion of the referee to determine if the spirit of this rule has been met. Students must point out to the referee when the robot uses one of the methods!

Meaningful use of line following to sucessfully complete one or more missions. (10 pts)

No | Yes

M08 Methods Used BONUS #2! To score points for this mission, the method used must be a meaningful part of the scoring solution. For example, squaring up on a mission model or wall to line up the robot before scoring points would be meaningful but squaring up on a mission model or wall after all scoring has been completed would not be. It is left to the discretion of the referee to determine if the spirit of this rule has been met. Students must point out to the referee when the robot uses one of the methods!

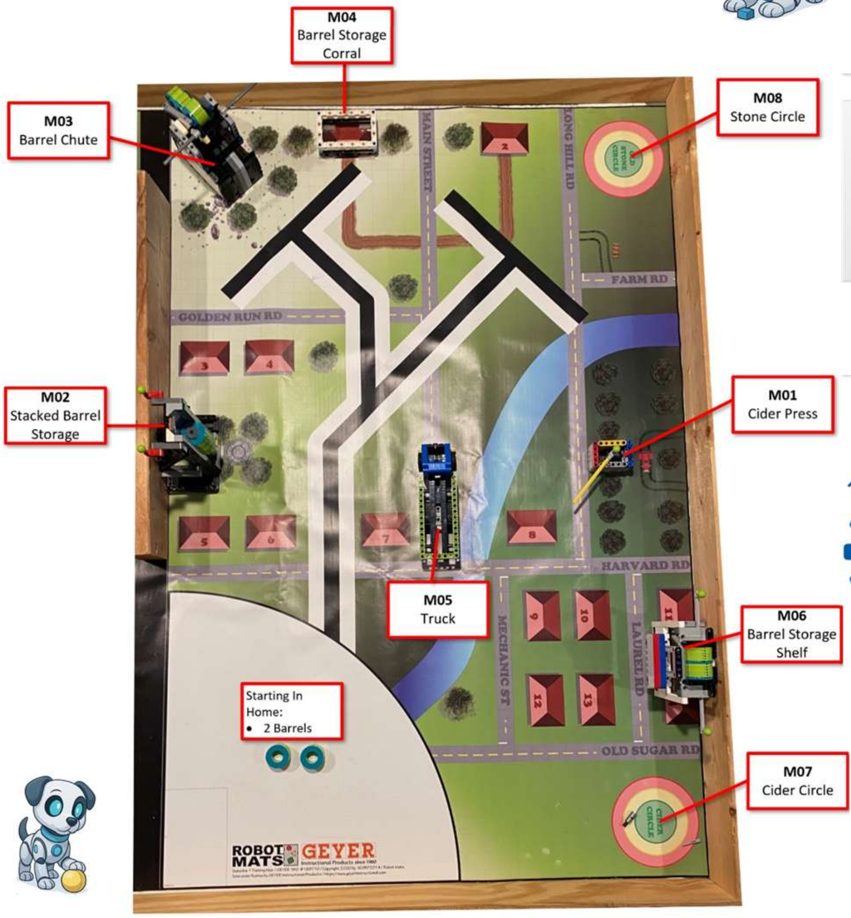
Meaningful use of squaring up on a wall or mission model to successfully complete one or more missions. (10 pts)

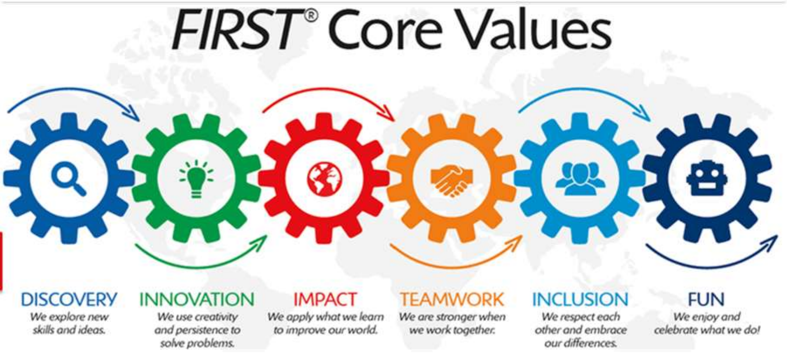
No | Yes

M08 Methods Used BONUS #3! To score points for this mission, the method used must be a meaningful part of the scoring solution. For example, squaring up on a mission model or wall to line up the robot before scoring points would be meaningful but squaring up on a mission model or wall after all scoring has been completed would not be. It is left to the discretion of the referee to determine if the spirit of this rule has been met. Students must point out to the referee when the robot uses one of the methods!

Meaningful use of wall following to successfully complete one or more missions. (10 pts)

No | Yes




FIRST® Core Values
DISCOVERY
We explore new skills and ideas.
INNOVATION
We use creativity and persistence to solve problems.
IMPACT
We apply what we learn to improve our world.
TEAMWORK
We are stronger when we work together.
INCLUSION
We respect each other and embrace our differences.
FUN
We enjoy and celebrate what we do!

Common Mistakes
1. Are you connected to the correct robot?
2. Did you download your code?
3. Are you running the new code?
4. Do you have any spelling errors or are there any spaces that shouldn't be there?
5. Are there any errors reported on the console?

Links:	
Lessons:	https://fssfll.github.io/fssfll/
Pybricks IDE:	https://code.pybricks.com/
Pybricks Docs:	https://docs.pybricks.com/en/stable/