

FTC TeleOp Lessons 1-7

Combined teacher answer key for the complete student workbook

TEACHER NOTE

Accept equivalent wording when the student can point to the matching code, telemetry, calculation, or safe observation. Never treat a live motor test as required evidence when a paper trace will answer the question.

How to use this key

- Pause first and let every student commit to an answer before revealing the expected response.
- Ask for code or observation evidence, not only a correct vocabulary word.
- For live tests, require the complete POWERING ON sequence and visible DISABLED confirmation before approach.

Lesson 1 - How an FTC TeleOp Program Works

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
Four main phases	INIT/setup, WAIT for PLAY, ACTIVE LOOP, and STOP/disabled.
Do declared variables control hardware?	No. A declaration creates a typed Java reference. <code>hardwareMap.get()</code> connects it to a configured device.
Why exact names?	The quoted code name must match the Robot Configuration name exactly, including case and underscore.
<code>waitForStart()</code>	Pauses <code>runOpMode()</code> until PLAY or a stop request.
Repeated actions	Read current joystick values, calculate/store power, send motor commands, publish telemetry, then check and repeat.
Why negate Y?	FTC gamepad Y is normally negative when pushed forward; unary minus turns that into a positive drive request.
Exit response	INIT maps hardware and reports status; <code>waitForStart</code> holds for PLAY; the active loop repeatedly reads, commands, and reports until STOP makes <code>opModelsActive</code> false.

Teacher look-fors

Students keep setup actions outside the repeating loop and move the program token back to READ only while active.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 2 - Meet the Motors

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
Declaration	Creates a DcMotor variable name; it does not connect to a motor.
hardwareMap job	Looks up a configured physical device and returns the matching DcMotor object.
Two arguments	Device type DcMotor.class and exact configuration name such as "left_motor".
Why reverse one motor?	Mirrored motors may spin the chassis in opposite physical directions for equal electrical commands; a direction setting aligns the forward convention.
Safe direction test	Secure on blocks, DISABLED, clear area, call POWERING ON, pause, visually confirm, use gentle power, STOP, confirm DISABLED, then approach.
Exit response	leftMotor is a Java variable/reference. "left_motor" is the exact Robot Configuration lookup key.

Teacher look-fors

Students distinguish a Java variable from a configuration name and never approach until STOP and DISABLED are confirmed.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 3 - INIT, Telemetry, and waitForStart()

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
What begins at INIT?	The SDK calls runOpMode(); hardware mapping and pre-start setup execute.
addData vs update	addData adds an item to the pending telemetry packet; update sends/displays the packet.
Does Initialized mean driving?	No. It shows setup reached that point; execution can still be waiting before the motor loop.
Before PLAY stick motion	The stick value can change, but setPower lines after waitForStart do not execute yet.
After STOP	opModelsActive becomes false, the loop exits, and the OpMode ends.
Exit response	waitForStart() is the gate: setup runs during INIT, PLAY releases the program into the active loop, and STOP makes the active condition false.

Teacher look-fors

Students separate centered sticks from STOP: one requests zero power; the other ends active execution.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 4 - The Active Loop

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
Loop condition	while (opModelsActive()) checks whether the OpMode should continue before each cycle.
One cycle	Check active, read inputs, calculate powers, set both motors, send telemetry, return to the condition.
Why reread inputs?	The driver can move the sticks at any time; each cycle must use the newest command.
Does 0.0 stop the loop?	No. It requests zero motor power while the OpMode remains active.
What prevents the next cycle?	STOP makes opModelsActive false at the next check, so execution exits the loop.

Teacher look-fors

Students trace the condition before the body and show that a fresh input is used each cycle.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 5 - Reading the Joysticks

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
double	A numeric type that stores decimal values such as motor power.
Assignment order	Evaluate the right side first, then store its result in the variable on the left.
Tank drive	Left stick Y commands the left motor; right stick Y commands the right motor.
Sign conversions	-1.0 -> +1.0; -0.7 -> +0.7; 0.0 -> 0.0; +0.35 -> -0.35; +1.0 -> -1.0.
Exit response	If right_stick_y is +0.35, rightPower is -0.35, requesting that side to move backward under the lesson convention.

Teacher look-fors

Students preserve magnitude while changing only the sign, and avoid assuming telemetry sign proves physical wheel direction.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 6 - Motor Power, Movement, and Telemetry

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
setPower parts	In <code>leftMotor.setPower(leftPower)</code> : object = <code>leftMotor</code> , method = <code>setPower</code> , argument = <code>leftPower</code> .
Sign and magnitude	Sign requests direction under the configured convention; magnitude requests relative effort/speed. 0.0 requests stop.
Power pairs	+/+ equal: straight forward request; -/- equal: backward; opposite signs: spin; unequal same signs: curve.
Telemetry purpose	Shows what the program commanded so the team can compare code intent with physical behavior.
Mismatch response	STOP, visibly confirm DISABLED, then verify telemetry/command, motor mapping, configuration name, wiring, and direction setting one fact at a time.
Exit response	First verify that the telemetry belongs to the intended side and value; then verify mapping/direction configuration before changing code.

Teacher look-fors

Students write a prediction first, compare it with telemetry and motion, and change only one cause after a mismatch.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.

Lesson 7 - Split Arcade Drive

Expected responses, evidence, and common misconceptions

PROMPT	EXPECTED RESPONSE / EVIDENCE
Input jobs	drive = -left_stick_y; turn = right_stick_x.
Mix formulas	left raw = drive + turn; right raw = drive - turn. Opposite turn signs create a speed difference or spin.
Why normalize?	Combined raw magnitude can reach 2.0. Divide both outputs by the same denominator so neither exceeds 1.0 and their steering ratio is preserved.
Cases	+0.5/0.0 -> +0.5,+0.5; 0.0/+0.5 -> +0.5,-0.5; +1.0/+1.0 -> raw +2.0,0.0, denominator 2.0, final +1.0,0.0.
Exit response	drive +0.6 and turn +0.2 gives left +0.8 and right +0.4. Denominator is max(0.8,1.0)=1.0, so no scaling is needed.

Teacher look-fors

Students show raw mixes before normalization and divide both sides by one shared denominator.

Safety correction to reinforce

If any person crosses the boundary, a component shifts, telemetry is missing, or motion differs from the prediction, the operator presses STOP. The team visibly confirms DISABLED before anyone touches the robot.