**Lesson Planning Template** **Teacher: DeSIRE**

| 📆Date: | Topic: Grab At A Certain Distance | Time Required: 90 minutes |
|---|---|---|

🎯**Learning Target/Objectives:**

- **I can** successfully compile and upload a robotic transport program that uses ultrasonic data to trigger specific movement sequences .
- **I can** modify RGB parameters and Action Group identifiers within source code to re-route autonomous transport destinations .
- **I can** interpret a program's "State Check" logic (isRunning) to explain how a robot ensures a previous task is finished before starting a new one.

📜**Vocabulary:**

- **Intelligent Transport**
- **Fixed Distance Grab**
- **isRunning() Function**
- **#define**
- **Millivolts (mV)**
- **Conditional Logic (if-else)**

❓**Guiding Questions:**

- How does the robot "decide" whether to move an object to the left, center, or right area of the map ?
- Why is the `myController.isRunning() == false` check vital for the mechanical safety of the robotic arm?
- In a real-world warehouse, what would be the advantage of using ultrasonic "Intelligent Transport" over a simple conveyer belt?
- How does changing the RGB values in the `ultrasound.Color` function help a human operator monitor the robot's progress ?

📚**Lesson Design Details:**

- **Activity 1: The Autonomous Logistics Audit**
  - **Focus:** Students place blocks at 6cm, 13cm, and 20cm. They must document the LED color and the final transport destination on a "Logistics Map" to verify the program matches the technical table .
- **Activity 2: The "Ghost" Destination Edit**
  - **Focus:** Students modify line 118. They must change the command from ACTIONGROUP_RIGHT to ACTIONGROUP_LEFT. They must predict and then verify that a "Red" detection now results in a "Left" transport .
- **Activity 3: Color-Coded Feedback Design**
  - **Focus:** Students redesign the visual alerts. They must modify the code so that all transport tasks glow "Green" while moving and only turn "Red" if they fail to detect a block.

**Key Points (Vocabulary):**

- **Intelligent Transport:** The process of a robot independently measuring an object's location and moving it to a designated area without human help .
- **Fixed Distance Grab:** A specific robotic behavior where the arm only activates its "grip" function once a precise distance threshold is met .
- **isRunning() Function:** A piece of code that checks if the robotic arm is currently in motion to prevent overlapping commands.
- **#define:** A programming command used to assign a user-friendly name (like ACTIONGROUP_LEFT) to a specific file number in the robot's memory.
- **Millivolts (mV):** The unit used by the controller to measure battery health, often monitored during long transport tasks.
- **Conditional Logic (if-else):** The "decision-making" part of the code that chooses between different paths based on sensory data .

## Key Points of Instruction

- **Integration of Sensors and Actions:** Explain that this lesson "marries" the distance reading from Lesson 7 with the movement groups from previous units. The distance is the "Cause" and the transport is the "Effect" .
- **Hardware Status Monitoring:** Introduce the concept of "Boolean Logic" through the `isRunning()` function. The robot asks a True/False question: "Am I busy?" If False, it proceeds to the next detect cycle.
- **Modular Code Names:** Show students line 8-10 in the code. Explain that using names like `ACTIONGROUP_RIGHT` makes the code readable for humans, even though the robot only cares about the numbers (13, 14, 15) .
- **Safety Protocols:** Remind students that the arm will move suddenly once an object is detected. Keep fingers away from the "movement zone" and do not cover the sensor for long periods .

**Teacher's Cheat Sheet**

| Technical Feature | Data Point / Required Action |
|---|---|
| **Sketch File** | distance.ino |
| **Memory Usage** | Approx. 23368 bytes (72% of storage) |
| **Red Range** | 5 to 7 cm |

| | |
|---|---|
| **Green Range** | 12 to 14 cm |
| **Blue Range** | 19 to 21 cm |
| **Update Frequency** | Every 250 milliseconds |
| **RGB Code Format** | (R, G, B, R, G, B) |

| Category | Standard Organization | Standard/Benchmark Code and Description |
|---|---|---|
| **Technology** | ITEEA | **STEL-2R:** Follow step-by-step instructions to safely use systems and troubleshoot common problems |
| **Computer Science** | NCSOS | **HS-CS-03:** Illustrate the ways computing systems implement logic, input, and output through hardware components |
| **Computer Science** | NCSOS | **HS-AP-14:** Create procedures with parameters to organize code and make it easier to reuse |
| **Digital Literacy** | ISTE | **1.5.d:** Students understand how automation works and use algorithmic thinking to develop a sequence of steps |
| **Engineering** | ITEEA | **STEL-2V:** Analyze the stability of a technological system and how it is influenced by components in the feedback loop |