Lesson Planning Template **Teacher: DeSIRE**

📆Date:	Topic: Password Lock	Time Required: 90 minutes

🎯**Learning Target/Objectives:**

- **I can** successfully configure the Arduino IDE to compile and upload complex "Password Lock" logic involving multiple hardware modules .
- **I can** explain how a robotic system uses an OLED screen to provide visual feedback to a user during a data-entry process.
- **I can** demonstrate how a robot arm performs conditional tasks based on whether a stored numerical value matches a new input .

📜**Vocabulary:**

- **OLED Screen Module**
- **State-Based Logic**
- **Reset Constraint**
- **Variable Storage**
- **Conditional Execution**
- **IDE (Integrated Development Environment)**

❓**Guiding Questions:**

- How does the OLED screen help reduce "user error" when setting a password on the robotic arm?
- Why did the engineers include a rule to reset the count to zero after 9 touches instead of letting it go higher?
- In a real-world security setting, what are the advantages of a robotic "physical lock" compared to a standard digital password on a computer screen?

📚**Lesson Design Details:**

- **Activity 1: Visual Feedback Verification**
  - **Focus:** Students set a 3-digit password (e.g., 2-1-4) and verify that the OLED screen correctly displays each digit as it is entered. They must document the "Reset" behavior by intentionally touching the sensor more than 9 times .
- **Activity 2: The Security Protocol Challenge**
  - **Focus:** Student A sets a secret password. Student B must try to "guess" the password. Students discuss how many possible combinations exist for a 3-digit password using digits 0-9.
- **Activity 3: Hardware Diagnostics Lab**
  - **Focus:** Students navigate the Arduino IDE to observe the "Memory Usage" prompt. They compare the size of this program (approx. 26040 bytes) to previous lessons to understand how adding an OLED screen increases the code complexity.

Key Points (Vocabulary):

- **OLED Screen Module:** A small digital display used to show text or numerical data (like the password) to the user.
- **State-Based Logic:** A programming concept where the robot behaves differently depending on current conditions (e.g., setting a password vs. entering a password).
- **Reset Constraint:** A safety feature in the code that resets the touch count to zero if it exceeds 9 consecutive touches.
- **Variable Storage:** The ability of the micro-controller to remember a specific password entered by the user.
- **Conditional Execution:** When a program only runs a specific section of code if a certain condition is met (entering the *correct* password).
- **IDE (Integrated Development Environment):** The software environment used to write and upload code to the robotic brain.

Key Points of Instruction

- **Module Synchronization:** Remind students that this program uses the Touch Sensor and the OLED Screen simultaneously. If the screen does not light up, they should check the hardware connections before re-uploading.
- **Data Limits:** Ensure students understand the "reset" rule. If they touch the sensor 10 times, the robot treats it as 0.
- **The "Take Back" Logic:** Explain that the robot is programmed to remember the object's original location. The second entry of the password triggers the "reverse" of the transport action.
- **Upload Stability:** Students must keep the USB cable perfectly still during the "verify" and "upload" phase to avoid a "Program upload failed" error.

Teacher's Cheat Sheet	
Parameter	**Technical Requirement**
Sketch Name	Password_lock.ino
Board Type	Arduino/Genuino Uno
Memory Usage	Uses approx. 80% of program storage space
Password Format	Three-digit numerical code

Input Rule	Continuous touches over 9 reset the count to 0
Action Sequence	Input Password -> Transport to Left -> Input Password -> Transport Back
Safety Warning	Do not move the USB cable during upload
Serial Connection	5V via USB cable

Category	Standard Organization	Standard/Benchmark Code and Description
Computer Science	NCSOS	**HS-CS-02:** Design and implement strategies for troubleshooting hardware and software problems .
Computer Science	NCSOS	**HS-AP-11:** Create a program that stores and manipulates data through the use of variables (the password) .
Technology	ITEEA	**STEL-2R:** Follow step-by-step instructions to safely use systems and troubleshoot common problems .
Engineering	ITEEA	**STEL-3H:** Optimize a system by identifying and managing various constraints (resetting count after 9 touches).
Digital Literacy	ISTE	**1.5.d:** Students understand how automation works and use algorithmic thinking to develop a sequence of steps .