**AP CSP CodeBot**

|  |  |  |
| --- | --- | --- |
| **MISSION 6 Obj 4-6 Line Follower** | | **Time: 45 minutes** |
| **Project Goal:** Students will use line sensor input to program the ‘bot to follow a line.  **Learning Targets**   * I can call the ls.check() function to use the ADC hardware channel scanning feature. * I can use logic operators to increase the reliability and speed of the ‘bot | **Key Concepts**   * The line follower ‘bot will need to continuously check for the presence of a line beneath all five sensors. * A list and a tuple are similar, but different. A list is mutable, or can change, and tuple is immutable, or cannot be modified. * Pre-coded functions from the botcore library can use the ADC hardware, making your program even faster! | |
| **Assessment Opportunities**   * Mission 6 Obj 4-6 Assignment * Submit the “LineFollower” program (part 1) | **Success Criteria**   * Improve the program with a center line sensor to keep the ‘bot straight | |
| **AP CSP Framework**  **DAT-2.D** Extract information from data using a program.  **DAT-2.E** Explain how programs can be used to gain insight and knowledge from data.  **AAP-1.C** Represent a list or string using a variable.  **AAP-2.F** Write expressions using logical operators and evaluate those expressions.  **AAP-2.H** Write conditional statements and determine their results.  **Computational Practice 2.B** Implement and apply an algorithm.  **Computational Practice 4.C** Identify and correct errors in algorithms and programs, including error discovery through testing. | **Materials**   * Several short tracks with lines for the ‘bots to follow. Each track should have a different curve. * Test Surfaces * Mission 6 Obj 4-6 Assignment / Answers * Solution code for LineSense\_obj6 | |
| **Teacher Notes**   * Objective 4: Students answer the questions and read the instructions in CodeSpace. They will start a new program file and use CodeTrek for the code. After they type the code, they will use the console panel to get readings. They will record the results in their assignment. Students should have a value for threshold that works for them before moving on to Objective 5. There are many values that threshold can be. * Objective 5: Students will read the instructions in CodeSpace, but modify the code using a function for drive(). To meet the validator for the goals, the else: branch will call the motors.run() function. Students should try the code on their own, but a sample solution is provided on the assignment. If you want to control when students get the solution, you can delete it from the assignment before distributing them. * Students experiment with their code and the ‘bot to see what happens on a variety of short courses. Before the lesson, prepare (or have students prepare) several short courses with a variety of curves. Students can adjust the speed in their code and try different courses to see what happens. They record their results on the assignment. * Objective 6: Students make another modification to their code. At this point they should call the drive() function in the last branch and not have two calls to motors.run(). The code solution is also provided. You can decide if you want students to see it, or if you remove the code snippet before distribution. * Students again experiment with their code. Use the same courses as before. Students can adjust the speed in their code and try different courses to see what happens. The ‘bot should stay on course better than before. * Students will continue to add code to this program. Decide if you want them to turn in the code for this assignment, or just continue until the end. | | |