**AP CSP CodeBot**

|  |  |  |
| --- | --- | --- |
| **LESSON: PT Practice #4** | | **Time: 45 minutes** |
| **Project Goal:** Students will complete a program that meets the Create PT requirements.  **Learning Targets**   * I can create and use a list in a meaningful way. * I can create a function with a parameter, selection and iteration. * I can use the parameter in an if statement. | **Key Concepts**   * The Create PT has specific requirements for the program students create. * The parameter doesn’t have to be used in an if statement, but it should have an effect on the functionality. The easiest way to do this is to use the parameter in an if statement. | |
| **Assessment Opportunities**   * CodeBot PT Practice #4 Activity Guide * CodeBot PT\_Practice4 program * (optional) Writing prompts | **Success Criteria**   * Create a list * Use the list in a loop * Create a function with a parameter * Create a function that has a loop and if statement | |
| **AP CSP Framework**  **CRD-2.A** Describe the purpose of a computing innovation.  **CRD-2.C** Identify input(s) to a program.  **CRD-2.D** Identify output(s) produced by a program.  **AAP-1.D** Develop data abstraction using lists to store multiple elements.  **AAP-2.H** Write conditional statements.  **AAP-2.K** Write iteration statements.  **AAP-2.N** Write expressions that use list indexing and list procedures.  **AAP-3.A** Write statements to call procedures.  **AAP-3.C** Develop procedural abstractions to manage complexity in a program by writing procedures. | **Materials**   * CodeBot PT Practice #4 slides * CodeBot PT Practice #4 Activity Guide / Answers * Code solution for final program * AP CSP Student Handouts * Create PT WR Prompts | |
| **Teacher Notes**   * This lesson will be completed on the computer, using CodeSpace for programming. * Use the Sandbox in CodeSpace for programming. This lesson is not part of a mission. * The activity guide can be distributed digitally. Space is provided for students to take notes during the programming. * Students will create a remix of LineFollow1 to meet the requirements of the Create PT. * The best experience will come from them modifying their own code. However, we want all students to be engaged, so you can give them the original code to remix if needed. * The LineFollow1 program is found in Unit 3 Mission 6 Obj. 7-8. * Follow the slides for instructions and guidance. Additional help is provided in the Teaching Guide below. * Solution code for the final program is provided. * *REQUIREMENTS NOTE: This program uses a tuple for vals. A tuple is an immutable list. It meets the requirements for a list according to the documentation in the student handouts. See side note →* | | |

**Teaching Guide**

**Warm-up (5 minutes)**

🧑‍🤝‍🧑 **Discuss** – Use a discussion strategy, like journaling, working at boards, selecting random students, or a form of think-pair-share. These are the same slides as the first two PT practice lessons, so it may take less time. Reviewing this many times should firmly cement the requirements. Hopefully by now the students fully understand the requirements.

* Slides 2-3
* Review the requirements for the Create Performance Task.
* Review how the Create PT is like a remix project.

The first question in the Coding section of the Activity Guide can be included in the warm-up.

**PT Practice 4 (30-40 minutes)**

💻 Students can work individually or with a collaborative partner.

**IMPORTANT!:** Students will refer to LineFollow1 from Unit 3 Mission 6. They need to have it completed and accessible. Alternatively, you can give students the code as a starter from the mission.

💡 **Teaching tip – Slides 4-5**

Students review the LineFollow1 program. Answer the first three questions on the activity guide.

💡 **Teaching tip – Slides 6-9**

The slides show how the program meets the requirements for creating and using a list.

💡 **Teaching tip – Slides 10-12**

The slides show how to modify the code so the calibrate() function has a parameter that is used in an if statement.

💡 **Teaching tip – Slides 13-15**

The slides discuss the need for a loop in the function, and different ways to add it. The code is not done for the student on this step. Two suggestions are given, and it is up to the students to decide how they want to incorporate a loop. The solution code shows both ways, one loop in each if statement branch.

Note: Students only need to add one loop, so they don’t have to put one in both if statement branches. But they can if they want to.

💡 **Teaching tip – Slides 16-17**

Students return to the activity guide and brainstorm their own remix ideas. Alternatively, you can use collaborative groups or whole class discussion with the question.

💡 **Teaching tip – Slides 18-20**

Final slides go over information about the program they may be asked to write about in the Create PT writing prompts. The wrap-up will ask potential writing prompts so students can practice their responses.

✅ **IMPORTANT!!**

Students should clear their CodeBot.

**Wrap-Up (5-10 minutes)**

The wrap-up reviews their programming process and gives potential writing prompts. The wrap-up questions are actual sample writing prompts from College Board.

If time permits, have students review each other’s responses. Or select a couple responses and anonymously display them with the class and discuss if the response answers the question.

Formative Assessment:

* Daily reflection or journal entry
* Wrap-up questions
* Completed program
* Exit ticket
* Optional – pull a writing prompt from the “Create PT WR Prompts” and have students write about their code.