**AP CSP CodeX**

|  |  |  |
| --- | --- | --- |
| **LESSON: PT Practice #1** | | **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**   * PT Practice #1 Activity Guide * PT\_Practice1 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**   * PT Practice #1 slides * PT Practice #1 Activity Guide / Answers * Code solution for final program   + PT\_Practice1 * 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 remix the Pixels1\_matrix program 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 last version of the Pixels1\_matrix program can be found in Traversing a List #2. * Follow the slides for instructions and guidance. Additional help is provided in the Teaching Guide below. * Solution code for the final remix is provided. * REQUIREMENTS NOTE: The function created by the student needs to have a parameter that has an effect on its functionality. This can be accomplished many ways. But students need to be able to understand how this works, and they may be asked to give two different function calls that run different parts of the function, or why it doesn’t run different parts of the function. The wording has changed a little bit the last couple of years. From a reader’s viewpoint, I can tell you that the easiest way for students to be able to do this is to use the parameter in an if statement, and for the if statement to be near the beginning of the function. All the examples in this unit will meet the requirement in this way. | | |

**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.

* Slides 2-4
* Review the requirements for the Create Performance Task.
* Review how the Create PT is like a remix project.

**PT Practice 1 (30-40 minutes)**

💻 Students can work individually or with a collaborative partner.

**IMPORTANT!:** Students will use the Pixels1\_matrix program from Traversing a List #2. They need to have it completed and accessible. Alternatively, you can give students the code as a starter from the traversing lesson.

💡 **Teaching tip – Slides 5-7**

Students review Pixels1\_matrix. Use the activity guide. Students should answer the first three questions before seeing slide 7. You can show it to help them with their answers, or wait until they answer the questions and let them check their answers with the slide.

💡 **Teaching tip – Slides 8**

Go over the slide and have students record their answers in the activity guide. As an option, you can use collaborative groups or whole class discussion with the question.

💡 **Teaching tip – Slides 9-10**

Students add another list to the program. An example is shown on slide 10, but the list can be anything the student creates, as long as it is a matrix and includes the pixel information. The list doesn’t need to be complicated, and it can be any length.

💡 **Teaching tip – Slides 11-12**

These slides outline the next step. First move the code from the main program to the function. The for loop that is already there needs to be indented again, including the sleep. Then use the code in this line before deleting it: 

The final code for this step is on slide 12.

💡 **Teaching tip – Slides 13-14**

These slides also go together, adding a parameter to the function and using the parameter in the code. The final code for the step is on slide 13.

💡 **Teaching tip – Slides 15-16**

Final steps of the program. The yellow star on slide 15 indicates the code for the program at this point is included. It is the final code.

💡 **Teaching tip – Slides 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**

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

✅ **IMPORTANT!!**

Students should clear their CodeX by running their “Clear” program.

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

The wrap-up reviews their programming process and gives potential writing prompts. 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.