**AP CSP CodeX**

|  |  |  |
| --- | --- | --- |
| **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**   * PT Practice #4 Activity Guide * 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**   * PT Practice #4 slides * PT Practice #4 Activity Guide / Answers * Code solutions for program steps   + PT\_Practice4\_part1   + PT\_Practice4\_part2   + PT\_Practice4\_part3   + PT\_Practice4\_final * 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 PT\_Practice2 and PT\_Practice3 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 PT Practice programs 2 & 3 are found in their lessons, earlier in this unit. * Follow the slides for instructions and guidance. Additional help is provided in the Teaching Guide below. * Solution code for three intermediate steps and the final program 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. 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 PT\_Practice2 and PT\_Practice3 from earlier lessons. They need to have it completed and accessible. Alternatively, you can give students the code as a starter from the earlier lessons.

💡 **Teaching tip – Slides 4-5**

Students review the last two PT Practice programs.

💡 **Teaching tip – Slides 6-9**

Students start the remix by creating a new file for the program. They will create a list of letters. They can be in any order, and the length of the list can vary. A function with a parameter is started, and the parameter is used in an if statement. The yellow star on slide 8 indicates the solution code up to this part is included (part 1).

💡 **Teaching tip – Slides 10-14**

Students add a for loop and another if statement to display the result of the game. The yellow star on slide 14 indicates the solution code up to this part is available (part 2).

💡 **Teaching tip – Slides 15-17**

The main program is completed to add button presses. The yellow star on slide 17 indicates the solution code up to this part is available (part 3).

💡 **Teaching tip – Slides 18-21**

These slides can be skipped, depending on the time left and the interests of your students. They show a goo =d example of procedural abstraction by managing complexity. And also it shows what is essential for the function requirements for the Create PT, and how the non-essential code can be put into separate functions, which will make writing about the code easier. The yellow star on slide 19 indicates the solution code for the final program is available (final).

💡 **Teaching tip – Slide 22**

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 23-24**

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. The last three questions on the wrap-up 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.