**AP CSP CodeX**

|  |  |  |
| --- | --- | --- |
| **LESSON: Lists #1** | | **Time: 45 minutes** |
| **Project Goal:** Students will evaluate code segments that involve lists and list methods.  **Learning Targets**   * I can define and identify parts of a list. * I can access an element of a list. * I can evaluate code segments with lists and list methods. | **Key Concepts**   * A list can be used when you have, or could have, many similar variables. * A list has elements, or items, organized in order by number. * The number used to organize the elements is called an index. * Data in a list can be changed, removed, inserted, and used in calculations. | |
| **Assessment Opportunities**   * Lists #1 Activity Guide | **Success Criteria**   * Define list vocabulary * Access and manipulate elements in a list * Evaluate list code segments | |
| **AP CSP Framework**  **AAP-1.C** Represent a list or string using a variable.  **AAP-2.N** Evaluate expressions that use list indexing and list procedures.  **Computational Thinking Practice 3.A** Generalize data sources through variables.  **Computational Thinking Practice4.B** Determine the result of code segments.  **Computational Thinking Practice 6.A** Collaborate in the development of solutions. | **Materials**   * Lists #1 slides * Lists #1 Activity Guide / Answers * Lists #1 Code Tracing Examples (to be printed) * Unit 2 Review and Test Questions * Optional: baggies and slips of paper for the lists mental model. | |
| **Vocabulary:**   * **List:** (from Mission 7) A sequence of items you can access with an index.   (from Lists Vocab) an ordered collection of elements   * **Index**: (from Mission 7) A number that keeps track of what choice should be displayed.   (from Lists Vocab) a common method for referencing the elements in a list or string using numbers   * **Element:** an individual value in a list that is assigned a unique index * **List Length:** how many elements it contains. Lists can grow or shrink as elements are added or removed. You can   calculate the current length by using the function: ***len(list\_name)*** | | |
| **Teacher Notes**   * This lesson is best with partners or in groups of three. * This lesson has “Group Code Tracing” Problems that should be printed for each student group. * The activity guide can be distributed digitally; it is used as a review after group work. * The programming concept of lists continues in the next lesson, where students are given more practice. * The teaching guide (below) gives the narration for one way to present the lesson and help students develop a mental model of lists. The beginning of the lesson uses slightly different list questions than the slides. Modify the slides (or lesson) as needed. | | |

**Teaching Guide**

**Warm-up / Design Process (5 minutes)**

This short warm-up is to introduce abstraction and functions. The assignment document is only used for the debugging table, and then at the end of the lesson, as a review.

💡 **Teaching tip – warm-up**

* In the last lesson (Mission 7) you were introduced to lists and how they can increase the functionality of a program. Today we are going to create a mental model of lists and practice working with list elements.
* What are some examples of lists in your every-day life?
  + Song playlists
  + Grocery or shopping lists
  + To-do lists
  + Phone contacts
* What is a variable? What are some limitations of variables? (can only hold one value at a time)

**Teacher-led mental model (20 minutes)**

💻 Students will NOT work at computers for this lesson. I recommend groups of three, standing at vertical white boards.

💡 **Teaching tip:**

Students can use small whiteboards or baggies (or other containers) to represent individual variables. I will use baggies as an example so modify as needed if the students use something else as their variable containers.

📓 **Preparation:**

Have students work in groups of three at vertical white boards. Each student will need one to three baggies and slips of paper to write on. Students at white boards may need tape to connect their baggies together. Students at magnetic boards will need magnets.

🧑‍🤝‍🧑 **Say: (Example #1)**

* Think about your favorite candy. Write it down on a slip of paper and put it in the baggie.
* These baggies represent individual variables. How many variables does your group need to represent all the favorite candies?
* You probably have more than one favorite candy. Write your other favorite candy on more slips of paper, and put each one in a separate baggie. Remember – each baggie, or variable, can only hold one value!
* If you want to write a program that uses all the favorite candies, how many variables would you need? What if you have more favorites? What if you change your mind and have less? How will these changes affect your code?
* All the variables are related. They could be called candy1, candy2, candy3, etc. Instead of creating all these individual variables, wouldn’t it be better to have ONE variable that could hold all the values? SURE! And we have this capability with lists.
* Tape your baggies together in a line to create a list.
* The order you taped your baggies is the order of the list. Now that we have a list, how can you access a single candy? We number them! Computers start counting by 0, so starting with 0, number each of your candies.
* The number of each candy is called an index, and the candy (or value) is called an element.
* How many elements are in your list of candies? Notice that the length of the list (number of elements) is one more than the last index.
* What would you call your list? What should its name be? The programming convention is to use a plural noun for lists. This is a visual clue that you are working with a list instead of a variable.

🧑‍🤝‍🧑 **Say: (Example #2)**

Students should continue in their groups. They can use baggies, white boards, etc. for their individual variables.

* Think of your favorite number. Write it on the white board.
* Now perhaps you have another favorite number. Write it on another white board.
* Think about how many variables you need to represent all these numbers.
* … continue this dialog similar to the first example. End with going over the definitions of list, index and element

🧑‍🤝‍🧑 **Show:**

* Display slides 10, 11 and 12. List Vocabulary and definitions.

**Group Practice (20 minutes)**

💻 Students will continue in their groups of three at the vertical white boards. Print the two Examples in advance and have them ready for students to work through.

💡 **Teaching tip:**

Students will have two examples to work through in their groups. They can do them in either order. They should compare their answers with other groups and discuss or re-work examples they do not agree on. The teacher can monitor and support as needed.

Answers to Examples:

* Example #A -- by groups Final alist = [4, 6, 1, 5, 6]
* Example #B -- by groups Final blist = [10, 9, 3, 7, 13, 1]

✅ Students worked in groups during this unplugged lesson. Then students work on the activity guide as a review of the lesson. The activity guide can be completed individually or in small groups.

Formative Assessment:

* Lists #1 Activity Guide
* Daily reflection journal or Google Form
* Class discussion on what they learned about lists
* Exit ticket

**SUCCESS CRITERIA:**

* Work in a group to complete example A and B
* Define list, index and element
* Access a single element in a list
* Manipulate values in a list by changing an element, adding elements, and removing elements