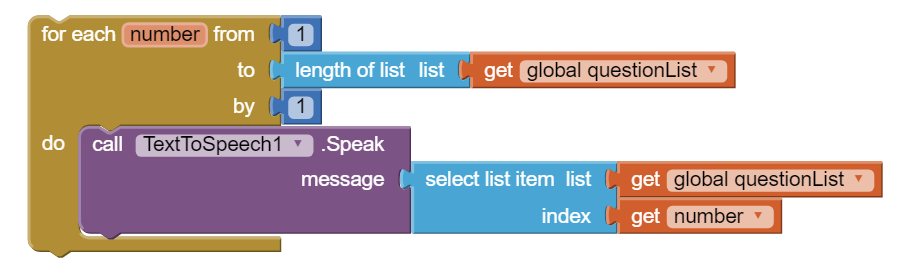
|  |  |
| --- | --- |
| In this lesson you will complete several small programming projects that add enhancements to the Quiz app. You are encouraged to discuss your ideas for how to solve these problems with the instructor and with your partner and other students.  **Objectives:** In this lesson you will:   * learn to count actions (right/wrong answers) using a list to keep track of which questions have already been answered, * learn to use loops with lists, * solidify your understanding of the quiz app through personalizing and customizing it. | [Click here to watch video.](https://youtu.be/g1CK3A9iFt4) |

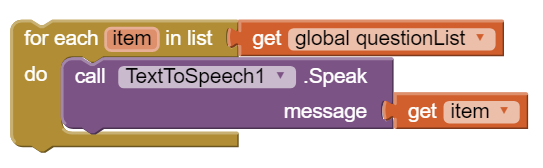
## 

## 

# Loops with Lists

Two main types of loops can be used to step through each item in a list. A “For each number Loop” uses a variable called number (often renamed as i) as the index of the loop; this variable starts at 1 and is incremented by 1 each time through the loop until it reaches the length of the list. In the following for loop, the “select list item” block is used to find the item at that index indicated by number and to speak it using the TextToSpeech block:



There is also a “For Each item in list” loop that selects each item in the list and assigns it to the variable item. This is simpler to use with lists but it does not keep track of the index. If you need the index, you can use a “index in list” block or use a for loop like above. You will need to use one of these loops with lists in the mini project #2 below.

In AP-style questions, the following pseudocode is used for a counting “for loop” that steps through a list with index i:

i ← 1

REPEAT n TIMES

{

DISPLAY( list[i] )

i ← i + 1

}

And the following pseudocode is used for a “for each item in list” loop:

FOR EACH item IN list

{

DISPLAY( item )

}

Notice that the square brackets [ ] are used to select a list item from the list using the index i. The following AP pseudocode is equivalent to the select list item block in App Inventor:

**questionsList[index]**  

A number or a variable can be put inside the square brackets to indicate the element number that you would like. For example, in the following code, we can refer to list item 1 using list[1] or using a variable i such as list[i].  
  
 list ← [ "item 1", "item 2" , "item 3" ]  
 list[1] = "new item 1" (Sets list item 1 to "new item 1")  
 DISPLAY(list[1)) (Displays list item 1)  
 i ← 2  
 DISPLAY(list[i]) (Displays list item 2 using i)

Some other list operations in AP-style questions are:

* INSERT(list, i, value) : inserts value into the list at index i, moving down all other items at and after i in the list.
* APPEND(list, value): adds value to the end of the list.
* REMOVE(list, i): removes the item at index i and moves up all items after the ith item.

# Getting Ready

Open App Inventor and open your Quiz app-- you’ll start with it. Or, if you don’t have a solution to the Quiz app, you can start App Inventor with the [Quiz App Projects Template](http://ai2.appinventor.mit.edu/?repo=templates.appinventor.mit.edu/trincoll/csp/unit6/templates/QuizApp/QuizAppProjectsTemplate.asc).

## 

# 

# Quiz App Mini Projects

## Mini Projects

Here are some creative projects. Complete all of them. **Use the *Save As* button to rename your project “QuizProject#” [where # will be replaced by the mini project number you will complete from the list of mini projects below.]**

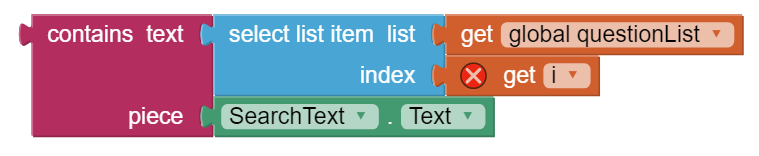
You are encouraged to discuss your ideas for how to solve these problems with the instructor and with your partner and other students.

1. **If/Else Scoring Algorithm:** Modify your app to keep score of how many questions are answered correctly and incorrectly. Be sure and restrict it so that the quiz taker can only receive credit for answering each question once (i.e., if there are three questions, the quiz taker can only be credited with three correct answers). Hint: define a list of true/false (booleans) values to record whether or not each question has already been answered correctly. You will need to use an *if* block to see if a question has already been answered correctly or not using this list before you update the score, and you will need to use the *select item from list* and the *replace item in list* blocks with this list.
2. **Loop Algorithm for Search:** Add a keyword search capability to your app. For example, if the user types in NASA and clicks on the search button, you should find the question or answer with the word NASA in it and show that question. This will be a linear search through the parallel question and answer lists using a loop. Here are the steps to this project:

* Add a TextBox for the user to type in a search word and a Button in a horizontal arrangement at the top of your app's UI.
* Refactor your code to put displayQuestion in a procedure that can be called from the search button, the next button, and from screen initialize once the index is set.
* When the search button is clicked, add a loop that steps through the lists like below using a loop counter i for the index of the loop.



* Inside the loop, you will need an if statement and contains text blocks from the Text drawer to test if the search word typed in by the user is in each question or answer.



* If the search word is in the question or answer, you should save the loop counter i into your global variable index and display the question at that index using your refactored procedure displayQuestion.

1. Use the Quiz App as a template to create a quiz on a topic of your own choosing. Besides changing the questions, answers, and pictures, add at least one enhancement to the app.

***Nice work! Complete the Self-Check Exercises and Portfolio Reflection Questions as directed by your instructor.***