3. WRITTEN RESPONSES

3 a.

3.a.i.

The purpose of the Parts of Speech app is to test users on their knowledge of the parts of speech that different words with different lengths correspond to.

3.a.ii.

The video shows how the user selects the length of the words and the part of speech that each word in the test corresponds to from different dropdowns. The buttons on the problemScreen screen make the next problem appear, while the other buttons set the screen to the following screen.

3.a.iii.

The user inputs a number into the program from the letterNumberDropdown dropdown. The output, a list of words that meet the chosen conditions, is displayed on the screen, one word per problem. The user inputs a string using the wordPartDropdown for each problem. The output, a list of words whose corresponding inputs were not the correct answers, is displayed on the resultsScreen screen.

3 b.

3.b.i.

```
4  var wordList = getColumn("Words", "Word");
5  var wordPartList = getColumn("Words", "Part of Speech");
```

3.b.ii.

```
47 function filterList(wordLength) {
48   if (wordLength == "any") {
49     filteredWordList = wordList;
50     filteredWordPartList = wordPartList;
51   } else {
52     filteredWordList = [];
53     filteredWordPartList = [];
54     for (var i = 0; i < wordList.length; i++) {
55       if (wordList[i].length == wordLength) {
56         appendItem(filteredWordList, wordList[i]);
57         appendItem(filteredWordPartList, wordPartList[i]);
58       }
59     }
60   }
61 }
```

3.b.iii.

On line 4, a list called wordList collects all of the words from the "words" database and stores them as a list. Each word is stored as a string. On line 5, a list called wordPartList collects all of the parts of speech corresponding to the aforementioned words. Each part of speech is stored as a string.

3.b.iv.

The wordList is a list of words used to create problems for the user to solve. It is filtered by the length of the words within it. Some words are then chosen from that filtered list, with each word used for one problem. These problems are then displayed to the user.

3.b.v.

Without the wordList list or wordPartList list, each word or part of speech would have to be stored in its own separate variable, which would each have to be checked to see if it met the requirements. The program code would increase by thousands of lines from less than 200. Code would have to be added or removed every time a word in wordList was added or removed.

3 c.

3.c.i.

```
47  function filterList(wordLength) {
48    if (wordLength == "any") {
49      filteredWordList = wordList;
50      filteredWordPartList = wordPartList;
51    } else {
52      filteredWordList = [];
53      filteredWordPartList = [];
54      for (var i = 0; i < wordList.length; i++) {
55        if (wordList[i].length == wordLength) {
56          appendItem(filteredWordList, wordList[i]);
57          appendItem(filteredWordPartList, wordPartList[i]);
58        }
59      }
60    }
61  }
```

3.c.ii.

```
31  onEvent("homeNextButton", "click", function( ) {
32    if (getText("letterNumberDropdown") == "Pick a number") {
33      setText("homeWarningLabel", "Please choose a valid input.");
34    } else {
35      filterList(getText("letterNumberDropdown"));
36      chooseWords();
37      // console.log(chosenWordIndex);
38      setText("letterNumberDropdown", "Pick a number");
39      userAnswers = [];
40      nextProblem();
41      setScreen("problemScreen");
42    }
```

3.c.iii.

The filterList procedure filters the wordList and wordPartList lists into the filteredWordList and filteredWordPartList lists, respectively, by finding words within the first and second lists whose length is the same number as the wordLength input. It is necessary to find words that meet the conditions given by the user choosing a word length.

3.c.iv.

An if statement checks if the content of the parameter wordLength is the same as the word "any". If it is, filteredWordList and filteredWordPartList are made equal to wordList and wordPartList, respectively. If it is not, filteredWordList and filteredWordPartList are cleared and a for loop is used to traverse wordList and append items to filteredWordList if their length matches the number given in the parameter wordLength. While words from wordList are being appended to filteredWordList, the index of each item being added to filteredWordList is used to append the corresponding parts of speech from wordPartList to filteredWordPartList.

3 d.

3.d.i.

First call:

Suppose the user selected the word "any" from the letterNumberDropdown dropdown. The argument passed through the filterList function would be the string "any" with the parameter wordLength.

Second call:

Suppose the user selected any of the number options, let's go with 5, from the letterNumberdropdown dropdown. The argument passed through the filterList function would be the integer 5 with the parameter wordLength.

3 d.ii.

Condition(s) tested by first call:

On line 48, there is an if-then statement that checks whether or not the wordLength parameter is a string containing the word "any". The wordLength parameter is a string with the word "any", which matches the condition given by the if-then statement.

Condition(s) tested by second call:

On line 48, there is an if-then statement that checks whether or not the wordLength parameter is a string containing the word "any". The wordLength parameter is an integer with a value of 5, which does not match the condition given by the if-then statement.

3.d.iii.

Results of the first call:

Because the wordLength parameter is equal to the string "any", the conditions given by the if-then statement are met and lines 49 through 50 would run. Thus, the filteredWordList and filteredWordPartList lists would become equal to wordList and wordPartList, respectively.

Results of the second call:

Because the wordLength parameter is equal to the integer 5, the conditions given by the if-then statement are not met and lines 49 through 50 are skipped. Instead, lines 52 through 59 would run. Thus, the filteredWordList and filteredWordPartList lists would be cleared. Then, the wordList and wordPartList lists would be filtered by finding the index of numbers inside wordList whose lengths match the parameter wordLength, or the number 5. All of the words with 5 characters from wordList would be added to filteredWordList, while their corresponding parts of speech would be added to filteredWordPartList.