Session 6: Practice Problems

Concept Questions

1. Explain what a list is and how you might use one.

2. How do you get an item from a list?

3. What is the difference between using the `add` block and the `insert` block to put an item in a list?

4. Explain the `for each item in list` loop. Can you use a `for i = _ to _` loop to do the same thing.

5. What is a higher order function? Give an example of using one with a list.

Programming Assignments

Play with a list

Come up with a list of things in your daily life. Last week we looked at my pets. This week make it about your family, friends, music, movies, a hobby... What it is doesn't matter, you just need some things to put in a list, and add and remove from a list. The more the example *means* to you, the more you are likely to care about it.

1. Start with a script variable - named to hold your chosen items - and set it to an empty list.

2. Then use the `add` and `insert` blocks to put some items in your list. Try to get in about 4-5 items, but there's no need to go overboard with a whole music library for now.

3. Find the `for each item in list` loop (import tools). Use it to say each item from your list in order.

4. Repeat this using the old `for i = _ to _` loop. There are some instances where you need to use this type of loop even when working with a list, so it is important that you can write both.

5. Come up with a condition for removing items from your list. For example, maybe you remove items that start with A, or items that are less than 4 characters long. Come up with something you can actually determine with the information you have. (Removing pets over age 10 won't work if there is no way to identify the pet's age.)

6. Now try removing items from the list in each type of loop based on the results of that condition. Can you use either of for loops to do this? Both? None? Why or why not?

   If you *think* that you can, make sure that your list has two items (better yet, three) that should be removed that are next to each other. This is good test data for a scenario like this. We want to make sure we don't skip one.

7. Here is one solution that will work with the `for i = _ to _` loop. When you get a match and remove an item, also change the i variable by -1. This prevents skipping an item.

8. Create a second script variable and set it to a new empty list. Using the `for each item in list` loop, add the items that do **not** match your condition to the new list, and do nothing with the ones that do. After going through the entire list, set your original script variable to the new variable that is now holding your cleaned up list.

9. Think about the solutions offered in steps 7 and 8 for ways to remove items from a list. Both of these solution might "break the rules" or what you or others might have perceived as the rules. Did you think of these ideas before they were suggested? If so, think about what lead you to think of them. How could you explain that process to someone else. If not, what do you think kept you from it? What can you do to try to think more "out of the box" in the future?

10. Finally, try using the HOF `keep items such that ___ from list` block to filter the items that you want removed out of your list. Remember that these HOF blocks need to have an empty "hole" or input spot where your current list item will be used. Also remember that this block will **keep** items that match your condition, so to remove items, you will need to reverse your condition. (If you want to *remove* things that start with A, then you want to *keep* things that *do not* start with A.)

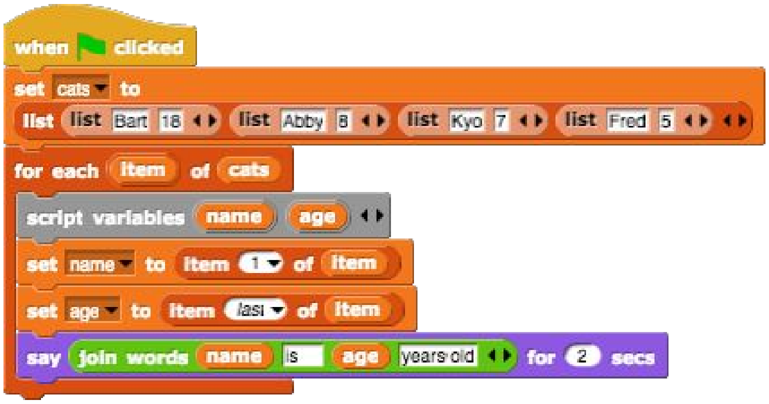
Make a To-Do List

Make a simple To-Do application. Have it ask for the things that you need to do, one by one. Make sure to give it a way to know when you are done entering them. Then have it tell you what your first to-do item is. Ask if it is completed. If it is, then remove it from the list, and go on to the next item. If not, you could either have it wait and ask again later, or just leave it and go on to the next item. This is your app, so you can make the call. When the to-do list is empty, have it congratulate you.

Use a Two-Dimensional list

In the first problem, Playing with a List, I was reminding you to select a condition that uses something you can actually determine from the information you have in the list. I gave the example that removing pets over age 10 won't work if there is no way to identify the pet's age. For this exercise, use a list in a list (or a two-dimensional list) to hold that extra data and use it to do something.

For example, here is info on my cats in a 2-dimensional list.



```
when 🏁 clicked
set cats to
list (list Bart 18 ◂▸) (list Abby 8 ◂▸) (list Kyo 7 ◂▸) (list Fred 5 ◂▸) ◂▸
for each (item) of (cats)
    script variables (name) (age) ◂▸
    set name to item (1 ▾) of (item)
    set age to item (last ▾) of (item)
    say (join words (name) is (age) years old ◂▸) for 2 secs
```