

Developing and Assessing New List Operators in App Inventor

Soojin Kim

Advisor: Franklyn Turbak

Designer

The image shows the MIT App Inventor 2 Designer interface. At the top, the header bar includes the MIT App Inventor 2 Beta logo, navigation links (Projects, Connect, Build, Help, Admin), and user information (My Projects, Guide, Report an Issue, Gallery, test@example.com).

Below the header, a green bar contains the project name "test" and buttons for "Screen1", "Add Screen ...", and "Remove Screen". On the right of this bar are tabs for "Designer" and "Blocks".

The main workspace is divided into four panels:

- Palette:** Contains categories like User Interface, Layout, Media, Drawing and Animation, Sensors, Social, Storage, Connectivity, and LEGO® MINDSTORMS®. Under "Drawing and Animation", there are icons for Ball, Canvas, and ImageSprite.
- Viewer:** Displays a preview of the app. It shows a status bar with a checkbox "Display hidden components in Viewer", a title bar "Screen1", and a canvas area with a small "Canvas1" component icon in the top-left corner.
- Components:** Lists the components used in the app, showing "Screen1" and "Canvas1". At the bottom are "Rename" and "Delete" buttons.
- Properties:** Shows the properties for the selected "Canvas1" component. Properties include BackgroundColor (White), BackgroundImage (None...), FontSize (14.0), LineWidth (2.0), PaintColor (Black), TextAlignment (center), Visible (showing), Width (Automatic...), and Height (Automatic...).

Blockly Editor

The screenshot displays the MIT App Inventor 2 Beta web interface. The top navigation bar includes the MIT App Inventor 2 Beta logo, a menu with 'Projects', 'Connect', 'Build', 'Help', and 'Admin', and a user profile section with 'My Projects', 'Guide', 'Report an Issue', 'Gallery', and a user email 'test@example.com'.

Below the navigation bar is a green header bar for the current project, labeled 'test'. It contains buttons for 'Screen1', 'Add Screen ...', and 'Remove Screen', along with 'Designer' and 'Blocks' tabs.

The main workspace is divided into two panels:

- Blocks Panel (Left):** A sidebar showing a categorized list of blocks. The categories are: Built-in (Control, Logic, Math, Text, Lists, Colors, Variables, Procedures), Screen1 (Canvas1, Button1, Label1), and Any component.
- Viewer Panel (Right):** The main area for building the app's logic. It shows a sequence of blocks:
 - Two 'initialize global' blocks: 'randomInteger' set to 0 and 'listofRandomIntegers' set to 'create empty list'.
 - A 'when Button1.Click' event listener block containing a 'do' block with the following actions:
 - 'set global randomInteger to random integer from 1 to 100'.
 - 'set Label1.Text to get global randomInteger'.
 - 'add items to list list' with 'item' set to 'get global listofRandomIntegers'.
 - 'get global randomInteger'.

At the bottom left of the workspace, there are warning and error counters (both showing 0) and a 'Show Warnings' button. A trash can icon is located at the bottom right of the workspace.

Problem with Current List Operators

Current List Operators



Map, Filter and Reduce

```
>>>(map (lambda x: x + 1) [5,3,8,10,2])  
[6,4,9,11,3]
```

```
>>>(filter (lambda x: x < 6) [5,3,8,10,2])  
[5,3,2]
```

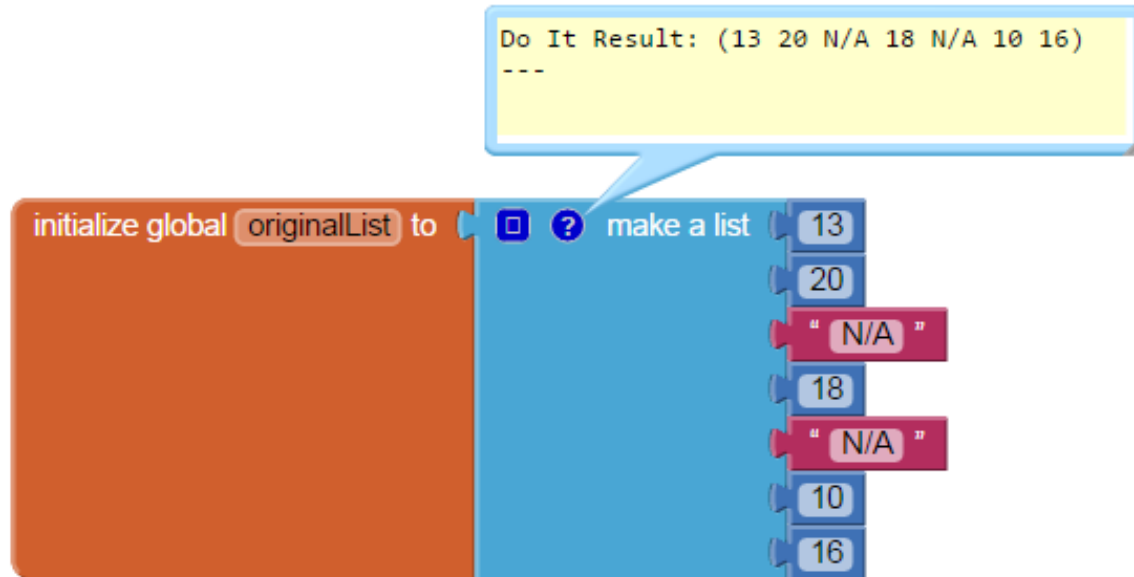
```
>>>(reduce (lambda x, y: x + y) [5,3,8,10,2])  
28
```

Berry's Lemonade

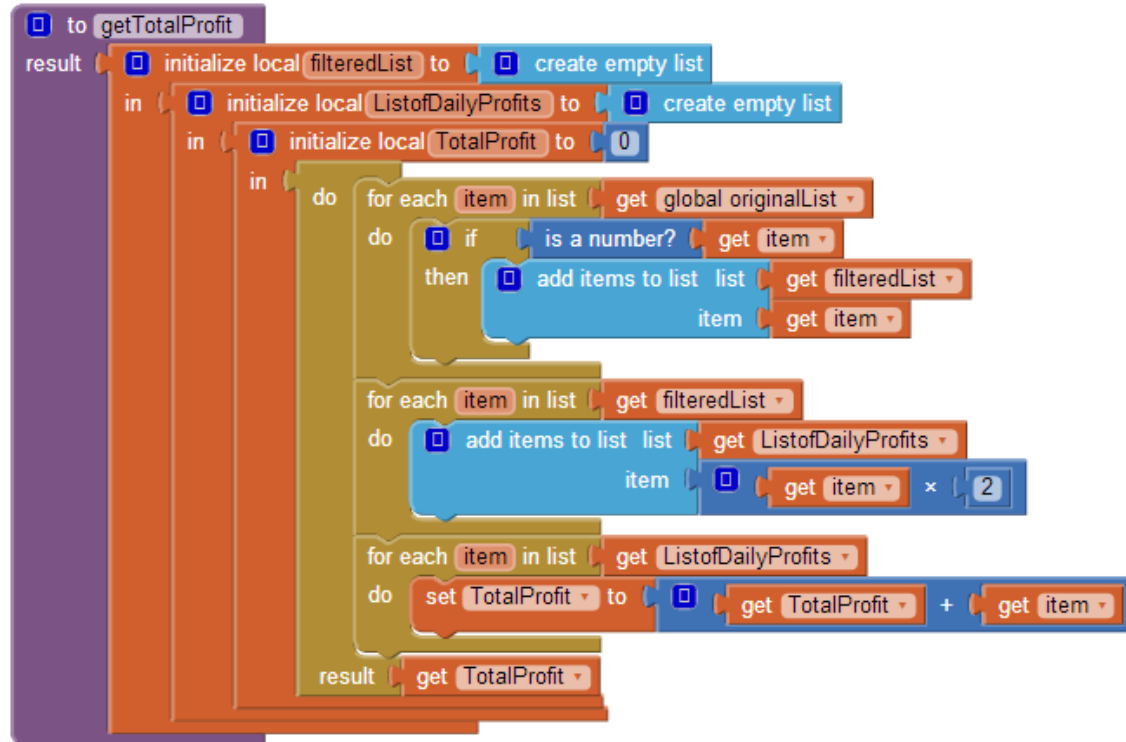
Date	# of lemonades sold	Daily Profit
6/1/13	13	
6/2/13	20	
6/3/13	N/A	
6/4/13	18	
.	.	
.	.	
.	.	
Total Profit:		?



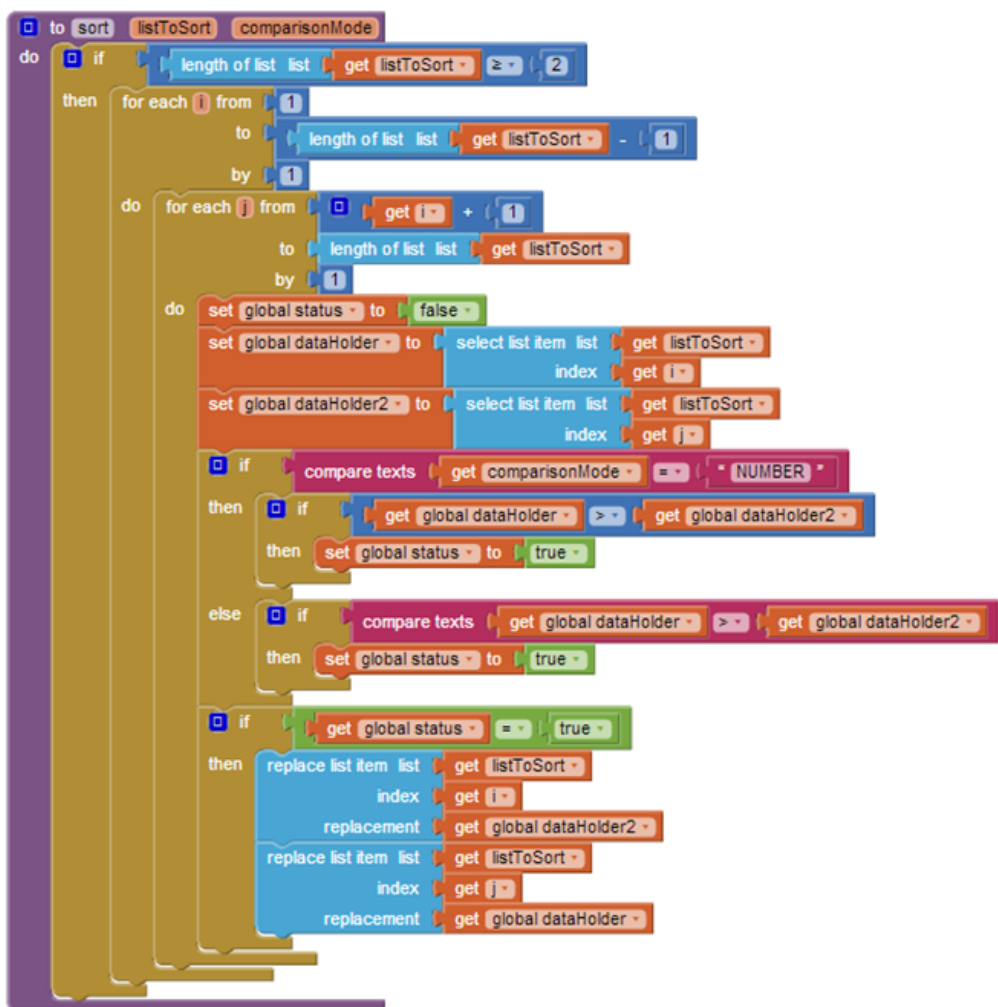
Old Design Using Loops



Old Design Using Loops



Sort: Old Design



<http://www.imagnity.com/tutorials/app-inventor/list-sorting-on-app-inventor/>

Solution: Addition of Higher-Order Operators

Map, Filter and Reduce

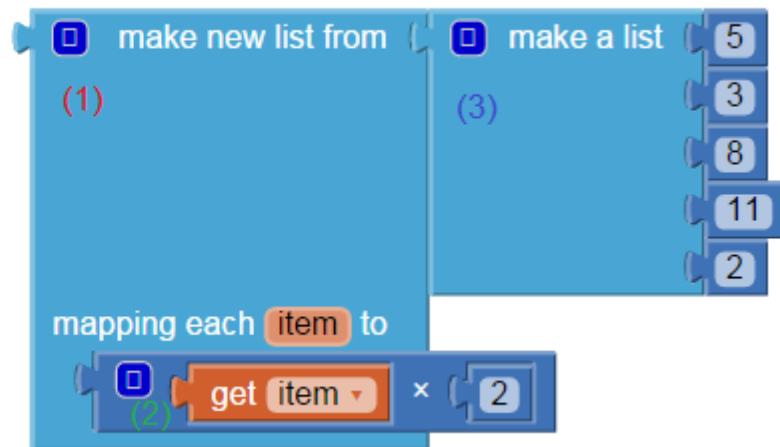
□ make new list from
mapping each `item` to

□ make new filtered list from
keeping each `item` passing
test

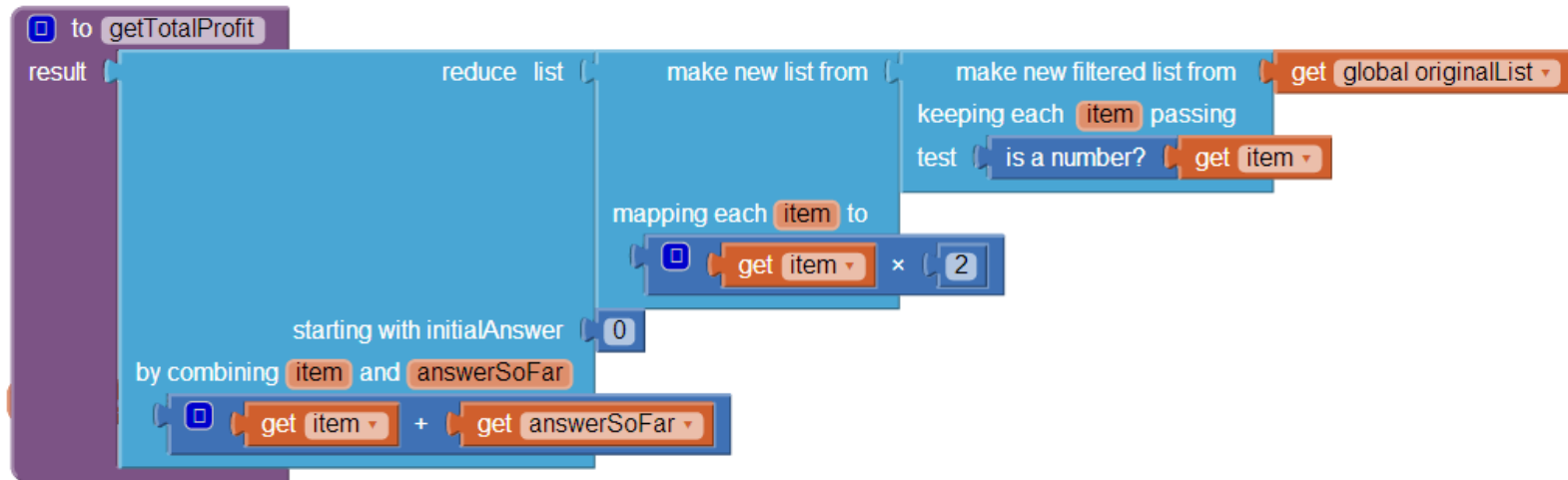
reduce list
starting with `initialAnswer`
by combining `item` and `answerSoFar`

Map Block

(1) (2) (3)
`map(lambda item: item * 2, [5,3,8,11,2])`



New Design Using Map, Filter and Reduce Operators



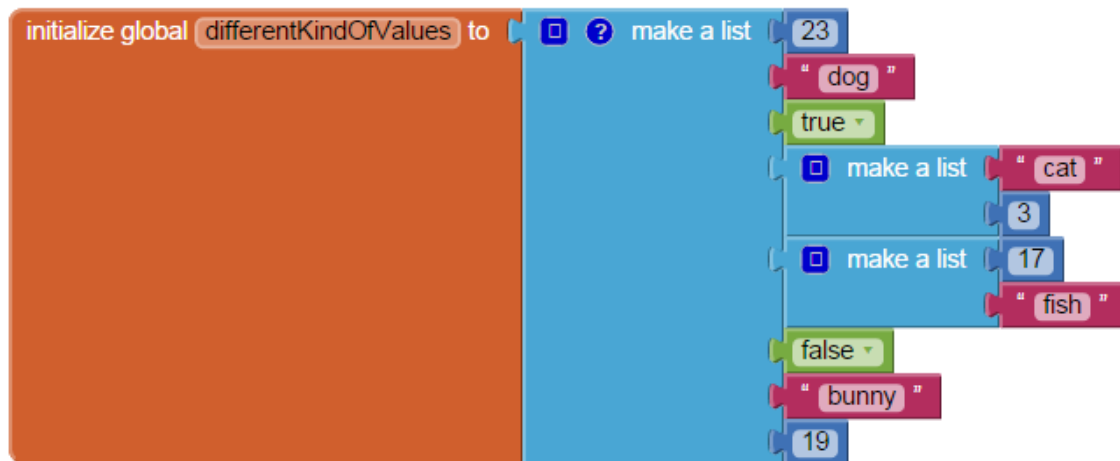
Three Sort Blocks

□ make new sorted list from

□ make new sorted list from
using key called on each `item`

□ make new sorted list from
where `item1` precedes `item2` if

Basic Sort



Do It Result: (false true 19 23 bunny dog (17 fish) (cat 3))

Do It Result: (23 dog true (cat 3) (17 fish) false bunny 19)

6 0

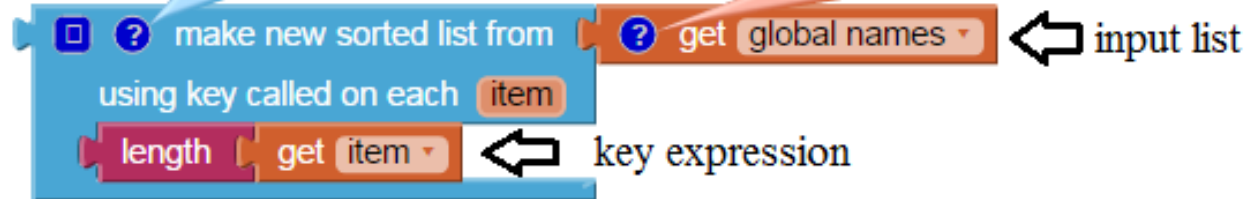
Show Warnings

`make new sorted list from` `get global differentKindOfValues` ← input list

Sort with Key

Do It Result: (Zoe Sam Alex David Megan Brendan)

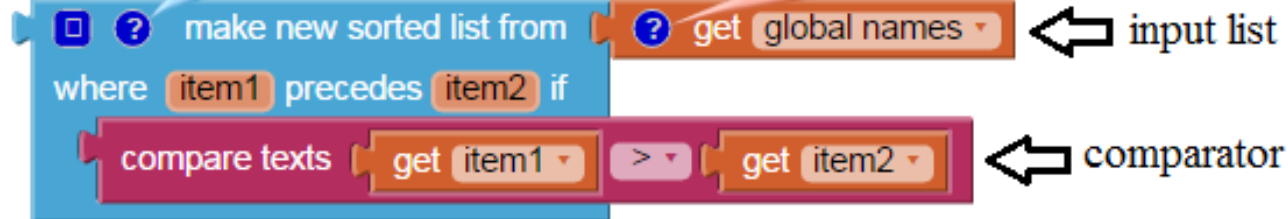
Do It Result: (David Brendan Zoe Sam Alex Megan)



Sort with Comparator

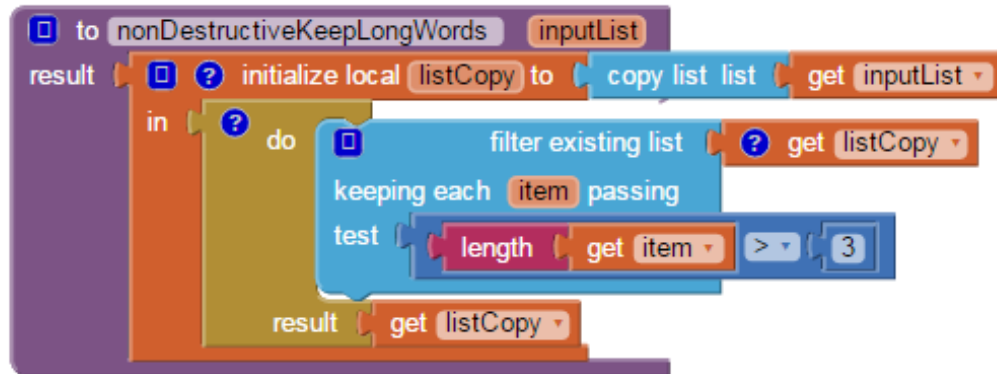
Do It Result: (Zoe Sam Megan David Brendan Alex)

Do It Result: (David Brendan Zoe Sam Alex Megan)



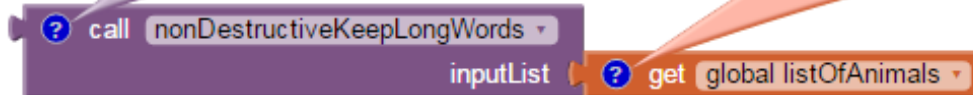
Destructive vs. Nondestructive Mechanism

Simulating Nondestructive Version of a Destructive Filter Operator

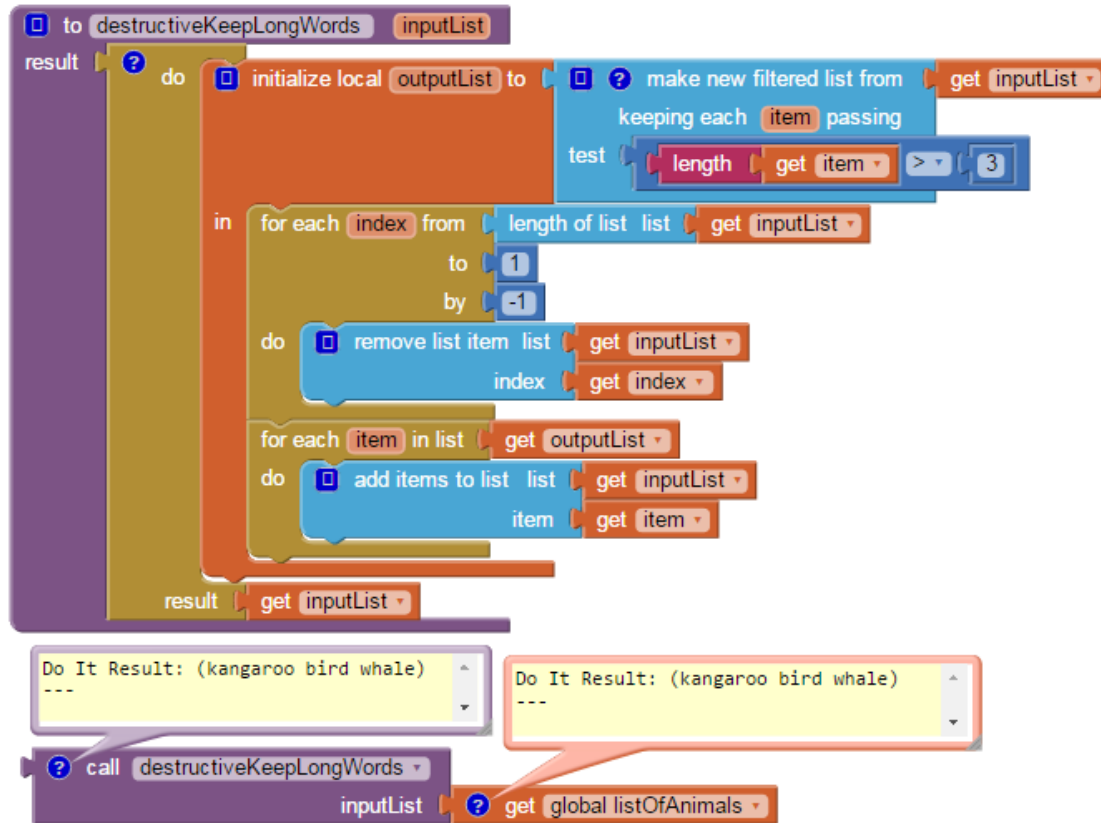


Do It Result: (kangaroo bird whale)

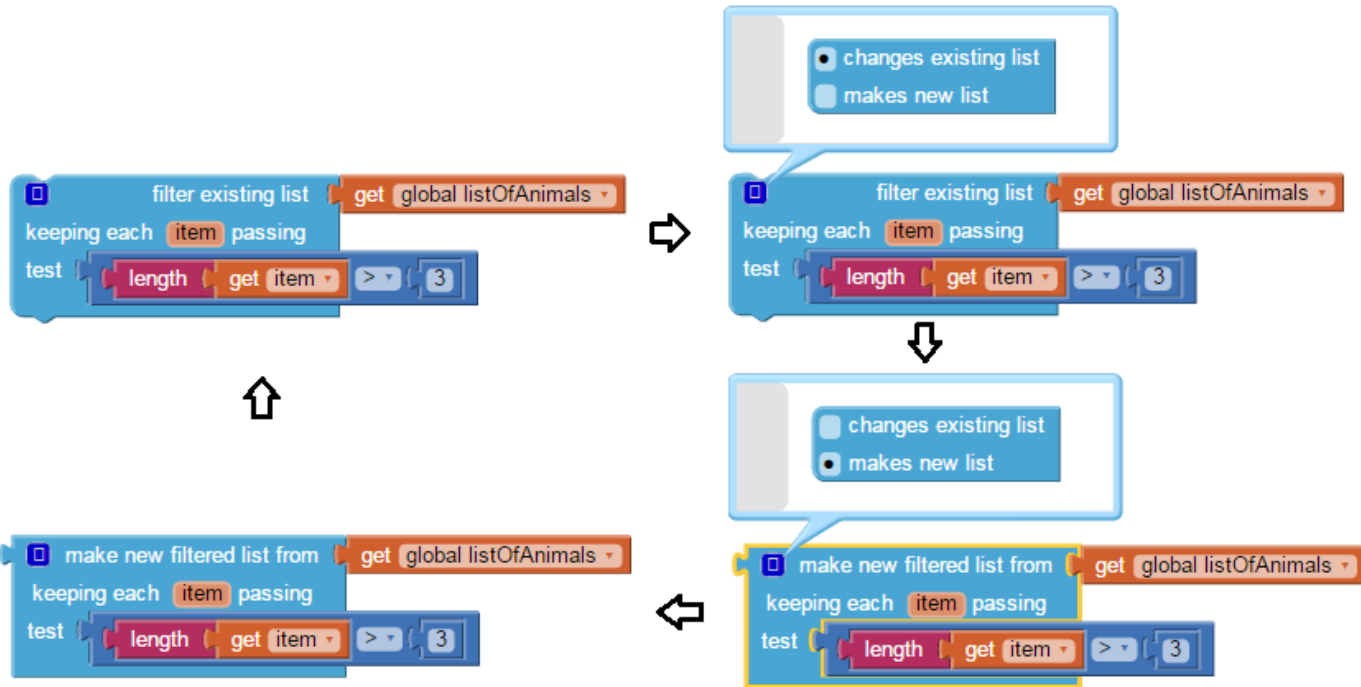
Do It Result: (dog kangaroo bat cat bird whale)



Simulating Destructive Version of a Nondestructive Filter Operator



Destructive vs. Nondestructive Mechanism



Design and Results of User Study

Design of User Study

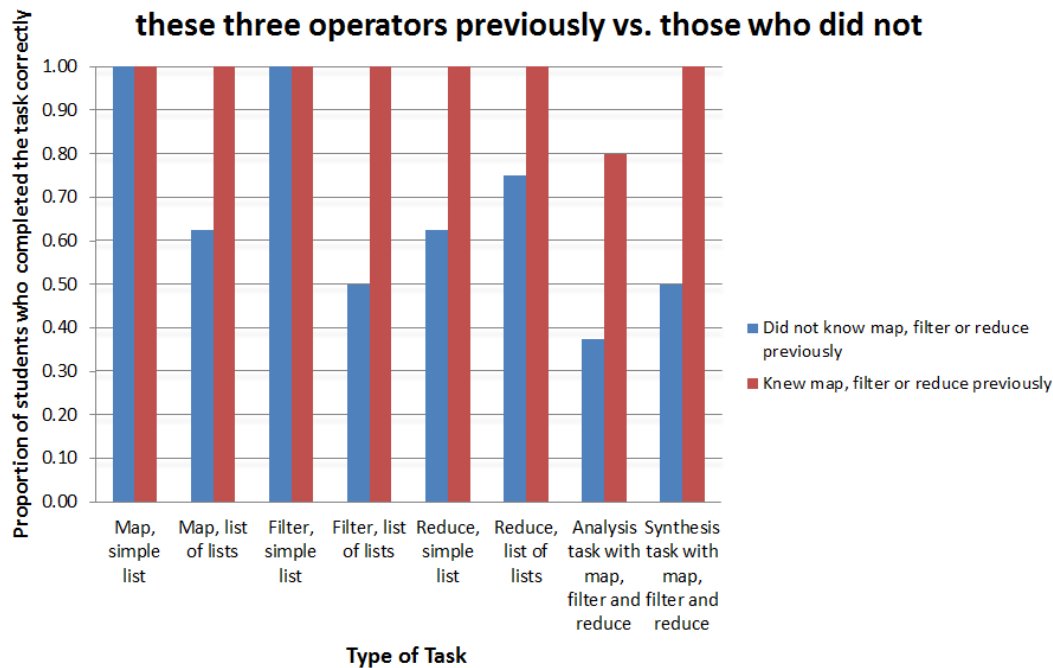
- Short tutorial on each list operator
- Part 1: 8 tasks involving mapping, filtering, and/or reducing
- Part 2: 6 tasks involving sorting

User Study Participants

- 18 Wellesley students who had previous experience working with App Inventor
- 10 users (56%) knew map, filter or reduce previously and the remaining 8 users (44%) did not

User Study Results Part 1

Comparison of performance on mapping, filtering and reducing tasks between students who knew at least one of these three operators previously vs. those who did not

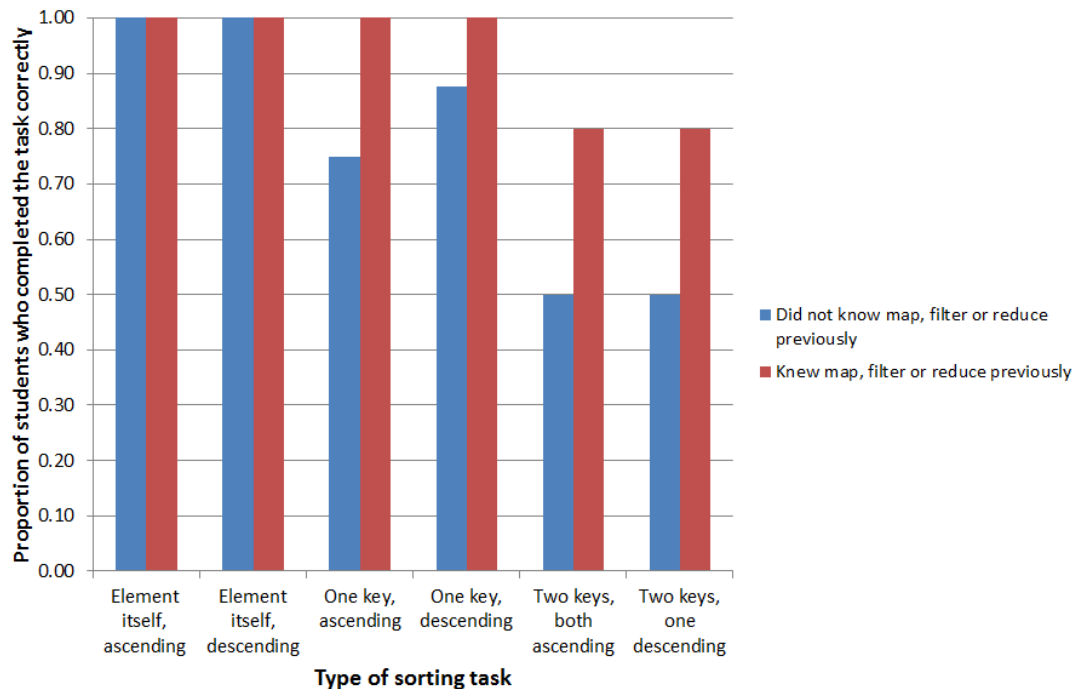


Feedback

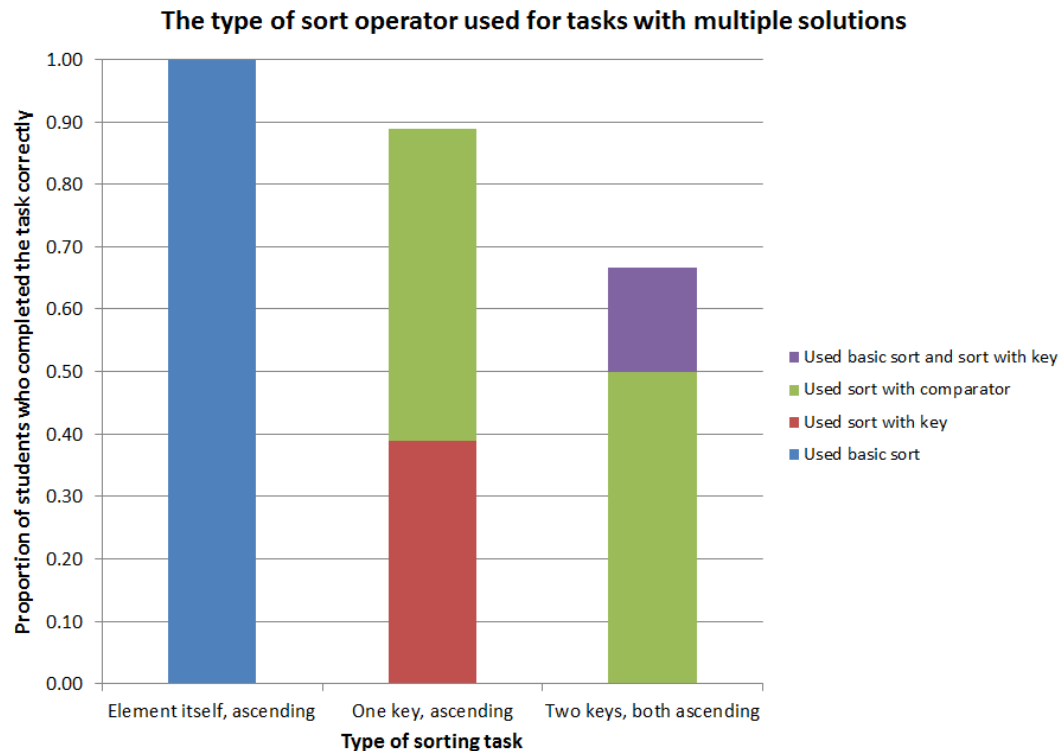
- “I have worked with map, filter, and reduce a lot in different languages, so the concepts were familiar and I was able to interpret pretty quickly what parts the blocks should have.”
- “These blocks were fairly simple to use, but I sometimes became frustrated because I would forget which block was useful in what kind of scenario. Reading the english on the blocks also helped with this though when I would get stuck.”

User Study Results Part 2

Comparison of performance on sorting tasks between students who knew map, filter or reduce previously vs. those who did not



User Study Results Part 2



Feedback

- “I didn't like that there were three different blocks for three different kinds of sorting...I almost think it would be easier if you had to explicitly decide how you want a list to be sorted every time you want to sort a list.”
- “I liked that there were three options, so I could use the one I felt most comfortable with.”
- “There are multiple ways you can perform a single task, especially with these three specific sort blocks. That made it both easier (can use any) and more difficult (many options for how to execute) to use.”

Future Work

- finalizing the labels on the blocks
- additional user studies
 - compare loops vs. higher order operators
 - test destructive vs. nondestructive mechanism
 - study wider demographic base