Discussion 3: Domain, Range, Lists and HOFs

SOLUTIONS

Domain and Range

1. Determine the domain and range of the following Snap! blocks:

   a. [contains block]
      
      Domain: First blank: list, second blank: any value
      Range: Booleans (True/False)

   b. [set var to foo block]
      
      Domain of foo: Numbers
      Range of foo: Booleans (True/False)
      Data type of var: Booleans (True/False), since the type of var is the same as the output type of foo

2. Fill in the table with the domain and range of the following higher order functions:

<table>
<thead>
<tr>
<th>Higher Order Function</th>
<th>Domain</th>
<th>Range</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>[map, over block]</td>
<td>First blank: reporter, with zero or more of its input slots left blank Second blank: list</td>
<td>List with same length as input list</td>
<td>~ The reporter must be able to take in all the data types in the input list without erroring. For example, if the input list to map has numbers and words, the input reporter must be able to handle numbers and words. ~ The output type of the reporter determines the data types of the values in the output list. For example, if the reporter outputs Booleans, then the output list will only have Boolean values.</td>
</tr>
</tbody>
</table>
### Higher Order Function Practice

1. Fill in the blanks so the `keep` block returns a list of the numbers from `MyList`.

   `MyList = [3, hello, goodbye, 5, 6]

   Keep items such that \[ \text{is a number} \] from `MyList`
2. Write an expression that returns the sum of the squares of the numbers in YourList.

YourList = [1, 2, 3, 4]

```plaintext
combine with + items of map x over YourList
```

3. Complete the following block so it works as described. Note: You may find the sentence -> list block helpful.

```plaintext
+ prepend every word + sentence +
```

```plaintext
report combine with join items of map join word over sentence -> list sentence
```

4. Describe (in words) what the following block outputs. Assume OurList is a list of words.

```plaintext
combine with join items of map letter (length of) of over OurList
```

It outputs a word consisting of the last letter of each word from our list.

**Challenge Problems**

1. Determine the domain and range of the following blocks:
a. Domain of Foo: **Booleans**, since the output of the equals block is passed in
Range of Foo: **A list of numbers**, since we know that the output of Foo is passed into the map block

b. Domain of Bar: **Not enough information**
Range of Bar: **Numbers**, since the output of Bar can be passed into the “>” block and can be compared with a number

2. If the output of Mystery is true, which of the following can you say for sure?

- [ ] A must be true
- [ ] B must be true
- [ ] A must be false
- [ ] B must be false
- [ ] None of these

If Mystery outputs true, then it could not have gone into either of the if statements, so A and B must be false.

3. You realize you could replace the *entire* body of Mystery with a single report statement. What could we report instead so that Mystery would have the same exact behavior? (Select all that apply)

- [ ] A and B
- [ ] A or B
- [ ] not A and not B
- [ ] not A or not B
- [ ] not A and B

To get this answer, we test each of the blocks with every possible combination of A and B (TT, TF, FT, FF) and find the ones that match the behavior of Mystery.

4. Given the following expression, what does NAMES evaluate to?

- [ ] a.
- [ ] b.
- [ ] c.
- [ ] d.

```
set NUMBERS to list 2 1
set NAMES to list Dan Garcia
for i = 1 to 2
  replace item i of NUMBERS with i
```
5. One of the most common data storage technologies is databases, think of them as Tables/Charts, with columns and rows. Let’s say you’re given a table that looks like the following:

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Height(inches)</th>
<th>Year</th>
<th>2 Favorite Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dan</td>
<td>75</td>
<td>4</td>
<td>[10, 61]</td>
</tr>
<tr>
<td>2</td>
<td>Mansi</td>
<td>65</td>
<td>4</td>
<td>[161, 10]</td>
</tr>
<tr>
<td>3</td>
<td>Bob</td>
<td>70</td>
<td>3</td>
<td>[70, 170]</td>
</tr>
<tr>
<td>4</td>
<td>Alice</td>
<td>71</td>
<td>1</td>
<td>[10, 160]</td>
</tr>
<tr>
<td>5</td>
<td>Nick</td>
<td>68</td>
<td>3</td>
<td>[161, 162]</td>
</tr>
<tr>
<td>6</td>
<td>Eve</td>
<td>64</td>
<td>2</td>
<td>[270, 370]</td>
</tr>
</tbody>
</table>

*Note: Each entry in “2 Favorite Numbers” is a list with exactly 2 numbers!*

You’re given a reporter block, \( \text{Column} \), that takes in the name of a column in the database and reports a list of all of the elements in that column in order. An example call is shown to the left.

For this problem, you can assume that you are only given the 3 HOFs, \( \text{Column} \), \( \text{map} \), and \( \text{combine} \), and any operators (green blocks) in Snap! For each of the subparts before, determine if the problem can be solved using only the given blocks.

a. Report the total sum of heights  
   Yes, we can use \( \text{combine} \) on \( \text{Column} \)(height)  
   Yes  No

b. Report a list of the names of people who are in year 4  
   No, our higher order functions cannot take in multiple lists. It can tell which years are equal to 4, but not which rows those 4s came from  
   Yes  No

c. Report the number you get when you multiply the squares of all of the heights above 70 inches together.  
   Yes  No

d. Calculate the sum of all of the numbers in the “2 Favorite Numbers” column  
   Yes  No
6. Indicate whether each set of blocks below is equivalent:

**a**
```
map (+ 2) over
keep items such that (>) 3 from list 2 3 4 4
```
No, the first block outputs [6, 6], and the second block outputs [4, 5, 6, 6]

**b**
```
map (+ 3) over map (× 2) over list 2 3
```
No, the first block outputs [7, 9], and the second block outputs [10, 12]

**c**
```
map (+ 3) over map (+ 6) over list 2 3
```
Yes! Both output [11, 12]

**d**
```
keep items such that (not) from list false true
```
No, this is a trick question! The second block will cause an error, because not cannot take in a list.