

Algorithms  
+  
Quest Review

Week 5 Practice Problems

# What is the Run-time?

```
Calculate Average of list
script variables index sum
set index to 1
set sum to 0
repeat length of list
  set sum to sum + item index of list
  change index by 1
report sum / length of list
```

The image shows a Scratch script for calculating the average of a list. The script starts with a 'Calculate Average of list' block. It then declares two script variables: 'index' and 'sum'. The 'index' variable is set to 1, and the 'sum' variable is set to 0. A 'repeat' loop is used to iterate through the list. The loop's length is determined by the 'length of list' block. Inside the loop, the 'sum' variable is updated by adding the value of the item at the current 'index' to the current 'sum'. The 'index' variable is then incremented by 1. After the loop, the final 'sum' is divided by the 'length of list' to calculate the average, which is then reported.

# What is the Runtime?



```
function DoubleSquare(list)
  script variables Index CurrentNum
  set Index to 1
  repeat length of list
    set CurrentNum to item Index of numbers
    replace item Index of numbers with
      CurrentNum + CurrentNum
    change Index by 1
  set Index to 1
  repeat length of list
    set CurrentNum to item Index of numbers
    replace item Index of numbers with
      CurrentNum * CurrentNum
    change Index by 1
```

The image shows a Scratch script for a function named `DoubleSquare` that takes a `list` as input. The script is organized into several blocks:

- Function Header:** A red block labeled `DoubleSquare` with a `list` input field.
- Script Variables:** A grey block for `script variables` containing `Index` and `CurrentNum`.
- Initialization:** A `set` block for `Index` to `1`.
- First Loop:** A `repeat` block for `length of list` containing:
  - `set` `CurrentNum` to `item Index of numbers`.
  - `replace item Index of numbers` with `CurrentNum + CurrentNum`.
  - `change Index` by `1`.
- Reset:** A `set` block for `Index` to `1`.
- Second Loop:** A `repeat` block for `length of list` containing:
  - `set` `CurrentNum` to `item Index of numbers`.
  - `replace item Index of numbers` with `CurrentNum * CurrentNum`.
  - `change Index` by `1`.

# What is the Run-time?

```
are the numbers of list distinct?  
script variables walter walker current  
set walter to 1  
repeat until walter > length of list  
  set current to item walter of list  
  set walker to walter + 1  
  repeat until walker > length of list  
    if current = item walker of list  
      say No, the numbers in the list are not distinct. for 1 secs  
      stop block  
    change walker by 1  
  change walter by 1  
say Yes, the numbers in the list are distinct. for 1 secs
```

The image shows a Scratch script designed to check if the numbers in a list are distinct. The script starts with a question block: "are the numbers of list distinct?". It then declares three script variables: "walter", "walker", and "current". The variable "walter" is set to 1. A "repeat until" loop is used to iterate through the list. The loop condition is "walter > length of list". Inside this loop, "current" is set to "item walter of list", and "walker" is set to "walter + 1". A second "repeat until" loop is nested inside, with the condition "walker > length of list". Inside this nested loop, an "if" block checks if "current = item walker of list". If true, it says "No, the numbers in the list are not distinct." for 1 second and then stops the block. If false, it changes "walker" by 1. After the nested loop, "walter" is changed by 1. Finally, after the outer loop, it says "Yes, the numbers in the list are distinct." for 1 second.

# What is the Run-time?

```
Mystery num list
script variables min max half current
set min to 0
set max to length of list + 1
repeat until max = min + 1
  set half to round (min + max) / 2
  set current to item half of list
  if current = num
    report half
  else
    if current < num
      set min to half
    else
      set max to half
report 0
```

The code is a Scratch script for finding the minimum element in a list. It starts with a function block named 'Mystery' that takes two arguments: 'num' and 'list'. Below this, it declares four script variables: 'min', 'max', 'half', and 'current'. The 'min' variable is set to 0, and the 'max' variable is set to the length of the list plus 1. A 'repeat until' loop is used to narrow down the search range. Inside the loop, 'half' is calculated as the rounded average of 'min' and 'max', and 'current' is set to the element at that index in the list. An 'if' statement checks if 'current' equals 'num'. If true, it reports 'half'. Otherwise, another 'if' statement checks if 'current' is less than 'num'. If true, 'min' is updated to 'half'; otherwise, 'max' is updated to 'half'. After the loop, it reports 0.

# What is the Run-time?



# What is the Run-time?

```
Mystery Func 2 list
script variables Index size result Index2
set Index to 1
set result to 0
set size to length of list
repeat 15
  set result to result + item Index of list
  set Index2 to 1
  repeat length of list / size - 1
    replace item Index2 of list with Index2
  change Index by 1
report result
```

The image shows a Scratch script for a function named "Mystery Func 2" which takes a parameter "list". The script uses several variables: "Index", "size", "result", and "Index2". It initializes "Index" to 1, "result" to 0, and "size" to the length of the "list". A main loop repeats 15 times. Inside this loop, "result" is updated by adding the value of the item at "Index" of the "list". Then, "Index2" is set to 1, and a nested loop repeats  $\frac{\text{length of list}}{\text{size}} - 1$  times. In this nested loop, the item at "Index2" of the "list" is replaced with the value of "Index2". After the nested loop, "Index" is incremented by 1. Finally, the "result" variable is reported.

# Quest Practice Problems



# Reading Practice 1 (Q4.Sp11)

*Blown to Bits* begins with a story of Tanya Rider who was rescued after 8 days of being trapped in a car. The book says you're supposed to say it's a story about "bits." IN ONE SENTENCE, what were the specific bits that saved her life?

# Reading Practice 2(Q4a.Sp11)

Match the programming paradigms to the facts that are best-suited.

- |                |   |
|----------------|---|
| a) Functional  | 1) “What” not “How”   |
| b) Imperative  | 2) Built through <i>composition</i> of blocks                 |
| c) OOP         | 3) Works via <i>message passing</i>                           |
| d) Declarative | 4) Like a <i>recipe</i> . Do this, then that, then that, etc. |

# Programming Practice 1 (Q3.F11)

Choose all that are not functions:

pick random  to

<

length of

sqrt  of

true

change  by

item  any  of

list

# Programming Practice 2 (Q6.Sp11)

**Question 6:** Suppose we wish to compute the mathematical function  $f(x) = 7 * (x - 5) * (x - 5)$ , and we have already been given the block definition on the right. Your job is to define the three helper blocks below so that `f` is computed correctly. Each of the blocks below should perform at MOST one computation of the form `num + num`, `num - num`, `num * num` or `num / num`. You don't have to draw images of the blocks, just the expression, e.g., `num + 99`



<p>A Scratch block for helper function <code>f1</code>. It has a yellow header with a green tab labeled <code>f1</code> and a red pin labeled <code>num</code>. The main body is green and contains the text <code>report</code> followed by a white rectangular input field.</p>	<p>A Scratch block for helper function <code>f2</code>. It has a yellow header with a green tab labeled <code>f2</code> and a red pin labeled <code>num</code>. The main body is green and contains the text <code>report</code> followed by a white rectangular input field.</p>	<p>A Scratch block for helper function <code>f3</code>. It has a yellow header with a green tab labeled <code>f3</code> and a red pin labeled <code>num</code>. The main body is green and contains the text <code>report</code> followed by a white rectangular input field.</p>
---	---	---

# Programming Practice 3 (Q6a.F11)

The function `foo` is used in the following way (and you have no idea what `a`, `b`, or `c` are). What can you say about the *domain* and *range* of `foo`?



Domain of `foo` (first argument):

Domain of `foo` (second argument):

Range of `foo`:

Login: cs10-\_\_\_\_\_

### Question 7:

You wish to author a command

**Sing Valentines Day Song** 

that takes as input a list of items, and has the sprite sing a song that describes what your true love gave you on that particular year (which, for year N, is N copies of the item in item N of the list, and all the things that were given last year). Example: If the list on the right were passed in as input, here's what song the sprite *should* sing to the user:



```
On anniversary # 1 of Valentine's day, my true love gave to me...
    1 partridge in a pear tree
On anniversary # 2 of Valentine's day, my true love gave to me...
    2 turtle doves
    1 partridge in a pear tree
On anniversary # 3 of Valentine's day, my true love gave to me...
    3 french hens
    2 turtle doves
    1 partridge in a pear tree
Whew! that's a lot of gifts!
```

a) What's the order of growth of **Sing Valentines Day Song** ? \_\_\_\_\_

Unfortunately, we have a bug in the code on the right:

b) Let's say we pass in the original 3-element list as input again. Indicate how the song above would change by crossing off whatever lines wouldn't be said (or changing them, or adding some, as necessary) in the box above.

c) Indicate (by writing directly on the code) a small change that fixes the bug.

