

CS 202: Introduction to Computation
Fall 2011: Practice Exam #2

Some of these questions were taken from a two-hour FINAL EXAM in a previous year, so your actual exam will be shorter than this practice.

Question 1: List your complaints here

Imagine you have eight different Scratch programs, each which manipulates a list “Mystery?” Your job is to “execute” the scripts in your head to repeat the exact same steps and operations that Scratch would. For each of the following 8 scripts, **show the contents of the List “Mystery” at the end of the script**. Be careful of the tiny but important differences across scripts!

```
when green flag clicked
  delete all of Mystery
  set Counter to 3
  repeat 5
    add Counter to Mystery
```

```
when green flag clicked
  delete all of Mystery
  set Counter to 20
  repeat 5
    add Counter to Mystery
    change Counter by 1
```

```
when green flag clicked
  delete all of Mystery
  set Counter to 1
  repeat 5
    insert Counter at 1 of Mystery
    change Counter by 2
```

```
when green flag clicked
  delete all of Mystery
  set Counter to 2
  repeat 6
    add Counter to Mystery
    change Counter by 2
  repeat 3
    delete last of Mystery
```

```
when green flag clicked
  delete all of Mystery
  set Counter to 2
  repeat 6
    add Counter to Mystery
    change Counter by 2
  repeat 3
    delete 1 of Mystery
```

```
when clicked
delete all of Mystery
repeat 5
  add 2 to Mystery
set Counter to 1
repeat 5
  replace item Counter of Mystery with length of Mystery
  change Counter by 1
```

```
when clicked
delete all of Mystery
set Counter to 2
repeat 10
  add Counter to Mystery
  change Counter by 2
set Counter to 1
repeat 5
  replace item Counter of Mystery with item Counter of Mystery + 1
  change Counter by 1
```

```
when clicked
delete all of Mystery
set Counter to 2
repeat 10
  add Counter to Mystery
  change Counter by 2
set Counter to 1
repeat 5
  replace item Counter of Mystery with item Counter + 1 of Mystery
  change Counter by 1
```

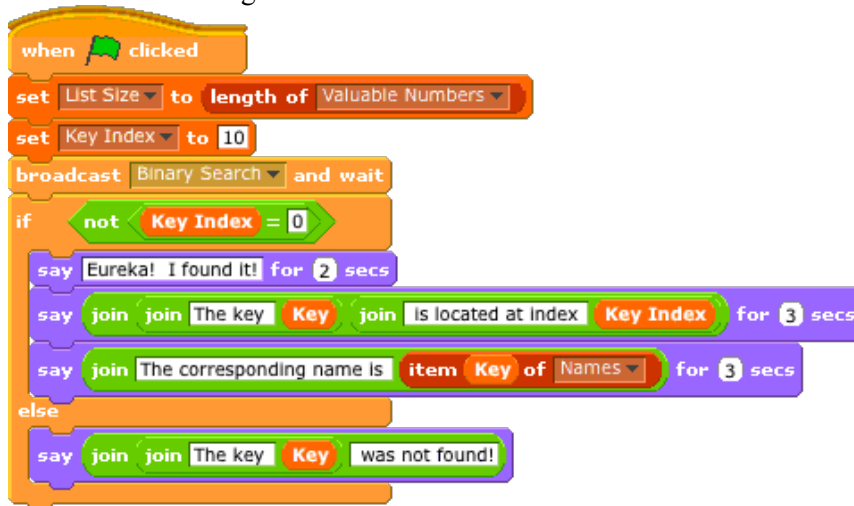
2) Searching for mistakes

The script shown in the Appendix implements a binary search. It accesses variables **Key**, **Key Index**, **hi**, **lo**, **index**, **Guesses**, and **List Size** and one list **Valuable Numbers**. The function **round x** rounds the number **x** to the nearest integer; numbers in the middle of two integers (e.g., 6.5) are rounded up (e.g., 7.0).

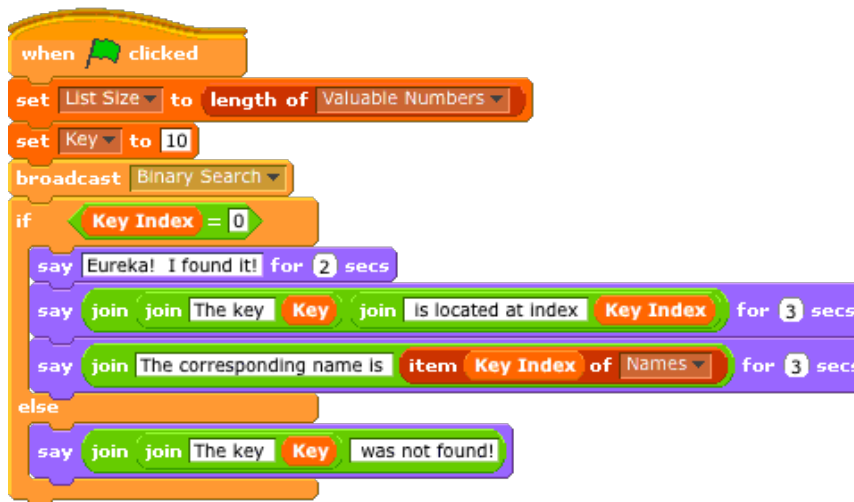
- A) You and a friend have the task of writing a Scratch program that uses the Binary Search script. Your program has access to an additional list called **Names** (shown in the appendix as well); each element of the Names list is associated with the corresponding element in the Valuable Numbers list. For example, the name Beth has a Valuable Number associated with it of 6 (since both are element 1 in their respective lists); the name Terry has a Valuable Number associated with it of 10 (since both are element 2 in their respective lists).

Your Scratch program should broadcast a message to the Binary Search script to find where the number **10** resides in Valuable Numbers, report that position (i.e., 2), and display the corresponding Name (i.e., Terry).

Your friend comes up with **two** possibilities for doing this. Unfortunately, each attempt has at least one problem – perhaps **even more than one!** Circle all the errors in each of the 2 scripts and show how the code should be changed..



```
when clicked
  set List Size to length of Valuable Numbers
  set Key Index to 10
  broadcast Binary Search and wait
  if not Key Index = 0
    say Eureka! I found it! for 2 secs
    say join join The key Key join is located at index Key Index for 3 secs
    say join The corresponding name is item Key of Names for 3 secs
  else
    say join join The key Key was not found!
```

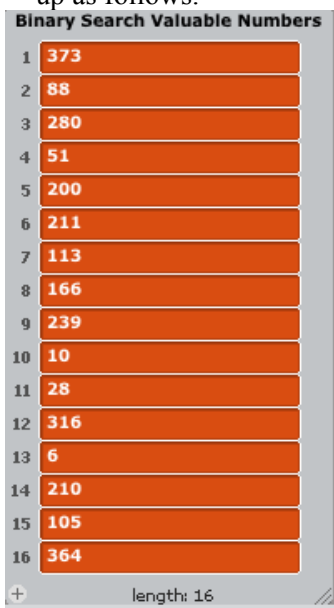


```
when clicked
  set List Size to length of Valuable Numbers
  set Key to 10
  broadcast Binary Search
  if Key Index = 0
    say Eureka! I found it! for 2 secs
    say join join The key Key join is located at index Key Index for 3 secs
    say join The corresponding name is item Key Index of Names for 3 secs
  else
    say join join The key Key was not found!
```

B) Assume you've figured out how to call the Binary Search script correctly. Using the data and script shown in the appendix, run the Binary Search script with **Key = 105**. Fill in the table below to show the values of each designated variable at the end of each iteration of the **repeat until** loop. You may not need all rows of the table.

Loop #	index	item	greater?	lo	hi	Key Index
1						
2						
3						
4						
5						

C) Imagine that something has gone terribly wrong with the Valuable Numbers list and it is now jumbled up as follows.



Imagine the Binary Search script is run again looking for Key = 105, but using this List as input. Fill in the table below to show the values of each designated variable at the end of each iteration of the **repeat until** loop.

Loop #	index	item	greater?	lo	hi	Key Index
1						
2						
3						
4						
5						
6						

D) What does the Binary Search script find in this case? Does the Binary Search script work correctly on this new data? Why or why not? (You should ignore any interaction with the Names list.)

3) Sorting through the Mess

The following Script “Sort List” uses a Selection Sort algorithm to sort a list of keys. Assume that this implementation is used for the following questions.



- 1) Assuming $N = \text{List Length}$ and using an input set of randomly ordered keys, the Selection Sort algorithm requires a number of steps that can be approximated as follows.
 - a. $O(\log_2 N)$
 - b. $O(N)$
 - c. $O(N \log_2 N)$
 - d. $O(N^2)$
 - e. $O(N!)$

- 2) Which of the following is the best description of the Selection Sort algorithm above?
 - a. It performs $\log_2 N$ iterations, each time finding the minimum key in the unsorted portion of the list and selecting it as the next key in the sorted portion of the list.
 - b. It performs N iterations, each time finding the minimum key in the unsorted portion of the list and selecting it as the next key in the sorted portion of the list.
 - c. It performs $\log_2 N$ iterations, each time finding the maximum key in the unsorted portion of the list and selecting it as the next key in the sorted portion of the list.
 - d. It performs N iterations, each time finding the maximum key in the unsorted portion of the list and selecting it as the next key in the sorted portion of the list.

- 3) Which variable in the script above records the separation point for the sorted and the unsorted portions of the list?
 - a. i
 - b. j
 - c. List Length
 - d. min
 - e. index of min

Imagine that the Selection Sort algorithm is run over a list of 10 keys and is stopped midway through the 10 iterations of the “repeat until” outer loop. Specifically, the outer loop runs only through iterations $i=1$, $i=2$, $i=3$, $i=4$, $i=5$, and stops just after the block “change i by 1” increments i to 6.

Assume you do not know the original contents of Unsorted List before the script was started; the list simply contains 10 integers between 0 and 100.

Consider each of the following lists of 10 keys. Is each list *possible* as the content for **Unsorted List** if the Selection Sort is stopped at this midway point?

To answer these questions, we believe you can use high-level understanding of how Selection Sort works, as opposed to the detailed specification given by the Scratch code.

Mark each list organization that is possible as True (a) and each list organization that is not possible as False (b).

4) 1 5 3 2 4 6 7 8 9 10

5) 1 3 2 9 5 4 6 8 7 10

6) 1 2 3 4 5 6 6 6 6 6

7) 1 2 3 4 5 10 20 15 29 8

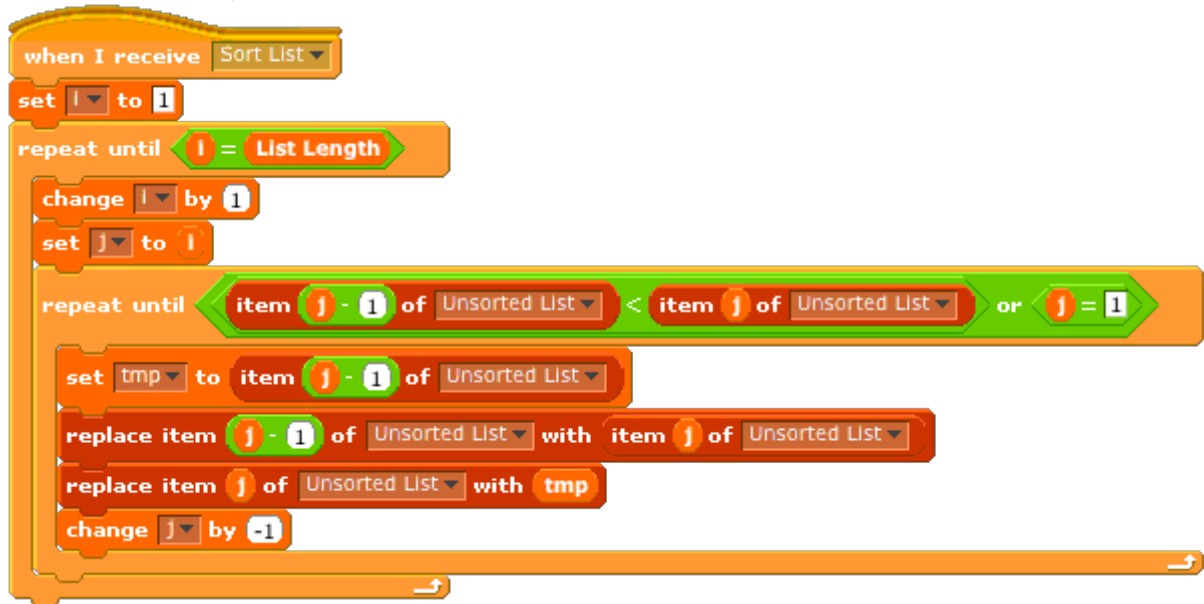
8) 1 2 3 4 5 6 20 38 17 25

9) 10 20 30 40 50 85 78 66 99 92

10) 10 20 30 40 50 25 35 45 55 65

11) 10 20 30 40 80 96 83 84 85 88

The following Script “Sort List” uses an Insertion Sort algorithm to sort a list of keys. Assume that this implementation is used for the following questions. (NOTE: This code is slightly different than what was shown in Fall 2011.)



```

when I receive Sort List
  set i to 1
  repeat until i = List Length
    change i by 1
    set j to i
    repeat until (item j - 1 of Unsorted List < item j of Unsorted List) or j = 1
      set tmp to item j - 1 of Unsorted List
      replace item j - 1 of Unsorted List with item j of Unsorted List
      replace item j of Unsorted List with tmp
      change j by -1
  
```

- 12) Assuming $N = \text{List Length}$ and using an input set of randomly ordered keys, the Insertion Sort algorithm requires a number of steps that can be approximated as follows.
 - a. $O(\log_2 N)$
 - b. $O(N)$
 - c. $O(N \log_2 N)$
 - d. $O(N^2)$
 - e. $O(N!)$

- 13) Which of the following is the best description of the Insertion Sort algorithm?
 - a. It performs $\log_2 N$ iterations, each time moving the next key in the unsorted portion of the list through the sorted list until it is greater than the key in the next lower position.
 - b. It performs N iterations, each time moving the next key in the unsorted portion of the list through the sorted list until it is greater than the key in the next lower position.
 - c. It performs $\log_2 N$ iterations, each time moving the next key in the unsorted portion of the list through the sorted list until it is less than the key in the next lower position.
 - d. It performs N iterations, each time moving the next key in the unsorted portion of the list through the sorted list until it is less than the key in the next lower position.

- 14) Which variable records the separation point for the sorted and the unsorted portion of the list?
 - a. i
 - b. j
 - c. List Length
 - d. tmp

The next set of questions examine the Merge Sort algorithm (not shown).

- 15) Assuming $N = \text{List Length}$ and using an input set of randomly ordered keys, the Merge Sort algorithm requires a number of steps that can be approximated as follows.
- a. $O(\log_2 N)$
 - b. $O(N)$
 - c. $O(N \log_2 N)$
 - d. $O(N^2)$
 - e. $O(N!)$

Merge Sort relies on the ability to merge two lists, List A and List B, into a single list. Which of the following are possible scenarios for merges within a correct Merge Sort? Mark each scenario that is possible as True (a) and each scenario that is not possible as False (b).

- 16) List A: 20 35 50 60 List B: 25 30 55 65 Merged List: 20 30 50 60
- 17) List A: 20 35 50 60 List B: 25 30 55 65 Merged List: 20 25 30 35
- 18) List A: 20 35 50 60 List B: 25 30 55 65 Merged List: 20 25 30 35 50 55 60 65
- 19) List A: 20 35 50 60 List B: 25 30 55 65 Merged List: 20 35 50 60 25 30 55 65
- 20) List A: 20 25 30 35 List B: 50 55 60 65 Merged List: 20 25 30 35 50 55 60 65
- 21) List A: 35 25 20 30 List B: 60 55 65 50 Merged List: 20 25 30 35 50 55 60 65

The next set of questions examine the Quick Sort algorithm (not shown).

- 22) Assuming $N = \text{List Length}$ and using an input set of randomly ordered keys, the QuickSort algorithm requires a number of steps that is can be approximated as follows.
- a. $O(\log_2 N)$
 - b. $O(N)$
 - c. $O(N \log_2 N)$
 - d. $O(N^2)$
 - e. $O(N!)$

- 23) Imagine that the following list of 10 keys is to be sorted by QuickSort.

3 9 5 6 0 2 7 8 4 1

Which key would be the best pivot for the first iteration?

- a. 3
 - b. 9
 - c. 5
 - d. 0
 - e. 1
- 24) Imagine that the following list of 10 keys is to be sorted by QuickSort.
- 20 35 61 18 95 82 44 58 5 79
- If key 79 is chosen as a pivot, how is the list likely to be organized after one iteration of QuickSort? (Choose one answer.)
- a. 20 35 61 18 95 82 44 58 5 79
 - b. 5 18 20 35 44 58 61 79 82 95
 - c. 5 18 20 35 44 58 61 79 95 82
 - d. 5 18 20 35 44 58 61 82 95 79
 - e. 20 35 61 18 44 58 5 79 95 82

4: Web Services and Security

- 25) The index of a web page search engine such as Google is used to perform which of the following mappings? (Choose one.)
- The index maps web pages to URLs.
 - The index maps web pages to IP addresses.
 - The index maps web pages to search terms.
 - The index maps web pages to web crawlers.
 - The index maps search terms to web pages.

For the next four questions, match each of the following goals of Information Security with its definition. Each definition a-d should be used exactly once.

- Can ensure that sender is who s/he claims s/he is
- Can ensure that message is not modified by anyone other than sender
- Can ensure that no one other than intended receiver can read message (i.e., no eavesdropping)
- Can ensure that message is actually received by receiver

- 26) Confidentiality (Match with a-d above)
27) Integrity (Match with a-d above)
28) Availability (Match with a-d above)
29) Authenticity (Match with a-d above)

- 30) You'd like to send the message "goodbye" to your friend using a Caesar cipher; you've agreed ahead of time to use a shift value of 4. What is the encrypted text that you will send?
- dbyegoo
 - hppeczf
 - ksshfci
 - No way to know without more information
 - None of the above

5) What is the meaning of Life?

Imagine you are asked to simulate the Game of Life. Remember, cells are placed on a 2-D grid; each cell can be either alive (black) or dead (white). The next generation of cells is calculated from the previous generation using a set of rules. For each cell, we can determine if it will be alive or dead in the next generation depending upon the current state of its 8 nearest neighbors (the 8 nearest neighbors are the cells directly adjacent above, below, left, right, and the four diagonal cells). We use the following rule:

If (cell is alive)

If < 2 neighbors are alive, then the cell dies

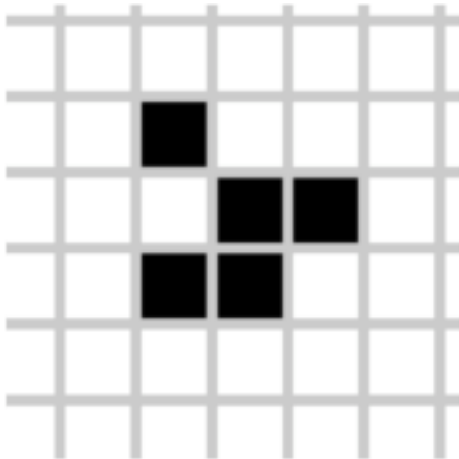
If > 3 neighbors are alive, then the cell dies

If 2 or 3 neighbors are alive, then the cell stays alive

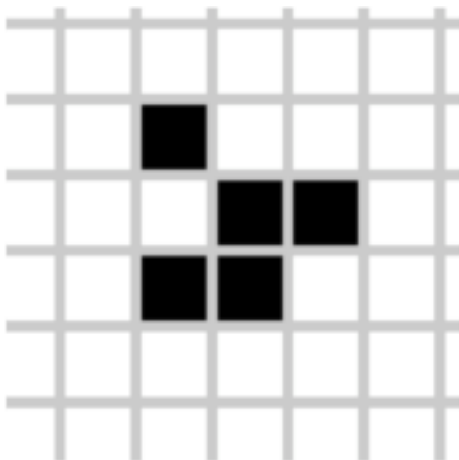
If (cell is dead)

if 3 neighbors are alive, then the cell becomes alive

Imagine the world begins in this initial state.



- A) On the grid above, show the number of the alive neighbors for each of the “interesting” cells; you do not need to report the number when there are 0 alive neighbors.
- B) On the grid below, show the state of the cells in the next generation. Alive cells should be filled in with black; dead cells should be left empty or crossed out with an X.



6) P vs. NP

Are each of the following problems considered P (a) or NP (b)? (Choose the one best answer.)

- a. P
- b. NP

- 31) A problem for which a known algorithm can find a solution in polynomial time
- 32) A problem for which a given solution can be checked in polynomial time, but no known algorithm exists for finding a solution in polynomial time
- 33) Sorting N integers
- 34) Searching through N unsorted integers for the maximum value
- 35) A problem for which the best known algorithm requires time $O(N!)$
- 36) A problem for which the best known algorithm requires time $O(N^2)$
- 37) A problem for which the best known algorithm requires time $O(N \log N)$
- 38) A problem for which the best known algorithm requires time $O(N^{1000})$
- 39) A problem whose only known solution requires enumerating all of the possibilities and every combination of the inputs
- 40) Finding a minimal spanning tree over a graph of vertices and weighted edges
- 41) Finding a solution to a travelling salesperson problem

Appendix for Question 3. Binary Search Script

The Binary Search script has access to several variables named **Key**, **Key Index**, **hi**, **lo**, **index**, **Guesses**, and **List Size** and one list named **Valuable Numbers**. The function **round x** will round the number **x** to the nearest integer; numbers in the middle of two integers (such as 6.5) are rounded up (7.0).

Your program has access to an additional list called **Names**; each element of the Names list is associated with the corresponding element in the Valuable Numbers list. For example, the name Beth has a Valuable Number associated with it of 6 (since both are element 1 in their respective lists); the name Terry has a Valuable Number associated with it of 10 (since both are element 2 in their respective lists).

The image shows a Scratch-like environment with two lists and a script. The first list, 'Binary Search Names', contains 16 names from Beth to Amln. The second list, 'Binary Search Valuable Numbers', contains 16 numbers from 6 to 373. The script implements a binary search algorithm.

Binary Search Names	Binary Search Valuable Numbers
1 Beth	1 6
2 Terry	2 10
3 John	3 28
4 Mike	4 51
5 Susan	5 88
6 Paul	6 105
7 Michael	7 113
8 Rachel	8 166
9 Armando	9 200
10 Doug	10 210
11 Kimberly	11 211
12 Randy	12 239
13 Arvind	13 280
14 David	14 316
15 Kathleen	15 364
16 Amln	16 373

```
when I receive Binary Search
  set Guesses to 0
  set Key Index to 0
  set hi to List Size
  set lo to 1
  repeat until lo > hi
    set greater? to 0
    set index to round (hi + lo / 2)
    set item to item index of Valuable Numbers
    change Guesses by 1
    if item = Key
      set Key Index to index
      stop script
    if item > Key
      set greater? to 1
      set hi to index - 1
    else
      set lo to index + 1
```