CS-141 Final Exam Review

Presented by the RIT Computer Science Community

October 19, 2016

http://csc.cs.rit.edu

Python Basics

1. Although most in the industry will be reasonably forgiving when it comes to coding by hand, you should still be comfortable with the syntax of your language of choice.

   (a) Identify and fix the line(s) that have invalid syntax in the following code:
   If you are unsure about a specific line, just write a brief explanation of the error. Assume that incorrect lines do not affect the validity of other lines which may depend on them.

```
1  include math
2
3  int a = 0
4  b = 0
5
6  def function(arg):
7      return (arg + 2)
8
9  c = "hello"
10 sum = b + math.sqrt(c)
11 total = a + (b ** 0.5)
```

   (b) Which of the following *are* keywords or standard functions in Python?

```
import     in     from     class     define     int     str     String
True     false     where     while     or     not     isinstance     new
print     range     float     char     bool     xor     sum
```

General Recursion

2. Sally is being plagued by an army of lookalike suiters, each of which presents her with an enticing but unbearable meal upon arrival. The dishes all smell amazing, however, so she can't help but try each one. A dish will never fail to disappoint her, but fortunately, some of the suiters shared recipes and created identical concoctions. Once Sally tastes a meal once, she can immediately smell out instances of the same dish and send their bearers away.

   (a) Write a Python function that, given a list of "dishes," returns a list of the rejected dishes in the order they failed to fool her. (e.g. $[1, 2, 2, 3, 3, 3]$ would return [2,3,3])

(b) If your solution to 2a was iterative, write it recursively; if it was recursive, write it iteratively.

(c) What is the time complexity of your approach? Why?

3. Write a **recursive function** that reverses a string (e.g. "Racecar" yields "racecaR").

4. Perform a substitution trace on `reverse('Doge')`.

Files/IO

5. Write a function that takes in a file name, and returns the average size of a word in that file. Assume the files will only have 1 word per line, for example:

   No
   soup
   for
   you!

   which has an average length of: 3.25
   Assume a function `len( str )` *which returns the length of a string is provided.*

Tail Recursion

6. Below is a function that, given a lower bound i, upper bound n, and function from integers to integers f, computes $\sum_{k=i}^{n} f(k)$.

```
1     def series_sum(i, n, f):
2         if(i == n):
3             return f(n)
4         return f(i) + series_sum(i + 1, n, f)
```

   (a) Invoke the function to compute $\sum_{k=1}^{5} k^2$.

   (b) Rewrite the `series_sum` function to be tail recursive.

   (c) What is the advantage of the new implementation?

Greedy Algorithms

7. Given that an algorithm is *greedy*, is it guaranteed to return an *optimal* solution?

Structures

8. For the sake of this question, you find yourself to be the head programmer under Kim Jong Un's glorious reign. It also just so happens that a nation-wide track meet is being held today. Thus, the glorious leader has demanded that you write a program to keep track of information relating to all track runners present at the event.

   (a) Write a struct called `TrackRunner` to keep track of each competing runner. You will need to store each runner's `name` (a string), `age` (an int), and `fastestTime` (an int).

   (b) Now write a function to make an individual `TrackRunner` struct.

   (c) The glorious leader has decided that, on this day, no runner named "Joe" may win gold. Given a list of `TrackRunner` structs, write the function `aWinnerIsYou(runners)` that returns the runner in the list `runners` with the fastest time whose name is not "Joe". Use this function to find the runner's name, age, and fastest time and print those results.

   ```
   def aWinnerIsYou(runners):
   ```

Stacks and Queues

9. How can you use a stack to check if an input string of exclusively parentheses, brackets, and braces is properly balanced? (e.g. "[{}]" is accepted, but "(()" and "[(])" are not.)
   Assume that you are provided with a `Stack` structure that has `push()`, `pop()`, and `peek()` functions.

Searching

10. Given the sorted list [1, 4, 9, 16, 25, 69, 420, 1337], write out the steps that a binary search would take to find the number 69.

Trees

11. (a) If we have a **balanced** binary search tree containing n nodes:
    i. What is its height?

    ii. How much time would it take to traverse to any of the leaf nodes of this tree?

    iii. What's the worst case search time for an unbalanced search tree?

    (b) How many leaves are in a **complete** BST containing n nodes?

    (c) In terms of runtime efficiency, what is the worst possible configuration (e.g. arrangement of nodes and their children) of a binary search tree? Sketch a small example of what such a configuration looks like.

    (d) Based on your answer above, what data structure is such a BST reminiscent of?

12. (a) Write a binary tree program that defines the following:

- a `TreeNode` struct with the given properties:
  - `data` (an object) : the data in this node
  - `left` : the `TreeNode` head of the left subtree (or `None`)
  - `right` : the `TreeNode` head of the right subtree (or `None`)
- a `Tree` struct with the given properties:
  - `size` (an int) : the size of the binary tree
  - `head` (a `TreeNode`) : the head node
- a recursive function `is_in_tree(head, element)` that determines whether the binary tree with the given head contains the specified element

(b) Give an example of how you could test your code.

Heaps

13. For a binary heap containing n elements, what is the maximum number of swaps occurring after an insert operation?

14. Given a node in an array-based binary heap at index i, where are the indices of both its children? What is the index of its parent?

15. Run a heap sort in reverse order on the following list: [3,5,1,3,2,7,9], showing the heap at each stage. Be sure to heapify the list first.

Sorting

16. Fill in the table for the asymptotic running time of each sorting algorithm.

	Best	Worst	Average
Merge sort			
Quicksort			
Heap sort			

17. Below is Python code for a function that performs an insertion sort and prints `data` after each iteration of the `for` loop.

```
1 def insertion_sort( data ):
2     for marker in range( 1, len( data ) ):
3         val = data[marker]
4         i = marker
5         while i > 0 and data[i-1] > val:
6             data[i] = data[i-1]
7             i -= 1
8         data[i] = val
9         print( data )
```

    (a) Write out what the function will print for the input list: [3,2,7,1].

    (b) What is the sort algorithm's time complexity?

18. In what scenario does Quicksort experience its worst-case time complexity? You may assume that we always pick the first element as the pivot.

19. What causes Quicksort to run so slowly on the input you describe in the last question?

20. In Quicksort, why should we select a random pivot value, rather than always pivoting on, for example, the first or last element?

21. Show the stages of a merge sort and a quicksort on the following list: [3,5,1,3,2,7,9]. Be sure to identify your pivot.