Quiz 2 (Solutions)

22C:80 Programming for Informatics
Monday, April 6th, 2009

Notes: The quiz is worth **6 points total**. To get partial credit you must show your work.

---

1. Consider the following function:

```
def test(n):
  sum = 1
  while(sum <= n):
    for i in range(0, n):
      printNow("hello")
    sum = sum*2
```

(a) How many times will `Hello` be printed when you call the function `test(4)`?

Solution: 12 times. [The outter loop executes 3 times and the inner (for-loop) executes 4 times for each execution of the outter loop]

(b) How many times will `Hello` be printed when you call the function `test(n)`? Your answer should be expressed as a function of n.

Solution: Approximately $log_2 n * n$ times. [The exact answer is $(\lfloor log_2 n \rfloor + 1) * n$ times.]

(c) Justify your answer in part (b) with a 1-2 sentence explanation.

Solution: The outer loop executes $log_2 n$ times, approximately, because sum starts off with value 1 and it takes sum roughly $log_2 n$ doublings to exceed the value n, at which teim the loop ends. The inner loop executes n times for each execution of the otuer loop, implying that the *printNow* statement is executed approximately $log_2 n * n$ times.

2. The code for the functions `partition` and `recursiveQuickSort` are given below:

```
def partition(C, left, right):
  p = left
  for i in range(p+1, right+1):
    if(C[i] < C[p]):
      swap(C, i, p+1)
      swap(C, p, p+1)
      p = p + 1

  return p

def recursiveQuickSort(C, left, right):
  if(left < right):
    p = partition(C, left, right)
    recursiveQuickSort(C, left, p-1)
    recursiveQuickSort(C, p+1, right)
```

(a) Suppose `C` is the list `[6, 3, 7, 1, 9, 2, 7, 6]` and you call `partition(C, 1, 5)`. How does `C` change as a result of this call? What is the value returned by `partition(C, 1, 5)`?

Solution: C becomes $[6, 1, 2, 3, 9, 7, 7, 6]$. The value returned by partition is 3 which is the index of pivot 3 after doing the appropriate swaps.
The value of C after each iteration is:

i=2:	$[6, 3, 7, 1, 9, 2, 7, 6]$
i=3:	$[6, 1, 3, 7, 9, 2, 7, 6]$
i=4:	$[6, 1, 3, 7, 9, 2, 7, 6]$
i=5:	$[6, 1, 2, 3, 9, 7, 7, 6]$

(b) Suppose that we inserted a "print statement" that prints (in one line) `C`, `left`, and `right`, just before the call to the `partition` function in `recursiveQuickSort`. What are the first three lines of output you will see when you call `recursiveQuickSort(C, 0, 7)`, with `C` being the list `[6, 3, 7, 1, 9, 2, 7, 6]`.

Solution:

C	left	right
$[\mathbf{6}, 3, 7, 1, 9, 2, 7, \mathbf{6}]$	0	7
$[\mathbf{3}, 1, \mathbf{2}, 6, 9, 7, 7, 6]$	0	2
$[\mathbf{1}, \mathbf{2}, 3, 6, 9, 7, 7, 6]$	0	1

3. Consider the variant of *job scheduling* problem in which you get paid by the hour for usage of your conference room. So, your goal would be to maximize, not the number of requests you accept, but the total amount of time for which your conference room gets used.

   (a) Given this new goal, the "ends first" greedy algorithm that repeatedly picks an available request with leftmost right endpoint is no longer optimal. That is, there is a collection of requests for which there is a schedule that yields more revenue to you than what you would get by using the schedule produced by the "ends first" algorithm. Show such a collection of requests, clearly showing what the "ends first" algorithm would produce and how it is possible to do better.

   Solution: Given $I = [[1, 10], [2, 20]]$, the "ends first" algorithm will pick $[1, 10]$ yielding 10 units fo revenue, whereas picking $[2, 20]$ would have yilded 19 units of revenue.

   (b) Devise an alternate greedy algorithm for this version of the job scheduling problem, that seems more reasonable to you. Describe it clearly in plain English or in pseudocode. You do not have to say anything about its correctness.

   Solution: Repeatedly pick an interval with greatest span (length). After each choice of an interval, delete all other intervals that overlap the chosen interval.

   [This is quite similar to all the greedy job scheduling algorithms we ahve considered, differing only in the criterea used to make the greedy choice.]