**Sam Lazrak**

**CS 303-2B Algorithms and Data Structures**

**Homework Assignment 8**

**3/21/18**

1. Work the following Exercises from Chapter 15 of the text:
   1. (6 points) Exercise 15.1-4, page 370.

Modify MEMOIZED-CUT-ROD to return not only the value but the actual solution, too.

* MEMOIZED-CUT-ROD

let r[0..n] and s[0..n] be a new array

for i = 0 to n

r[i] = -

(val, s) MEMOIZED-CUT-ROD-AUX(p, n, r, s)

j = n

print s[j]

j= j = s[j]

* MEMOIZED-CUT-ROD-AUX(p, r, n, s)

if r[n] > 0

return r[n]

if n == 0

q = 0

else q = -inf

for i = 1 to n

(val,s) MEMOIZED-CUT-ROD-AUX(p, r, n - i, s)

if q < p[i] + val

q = p[i] + val

r[n] = 1

r[n] = q

return(q,s)

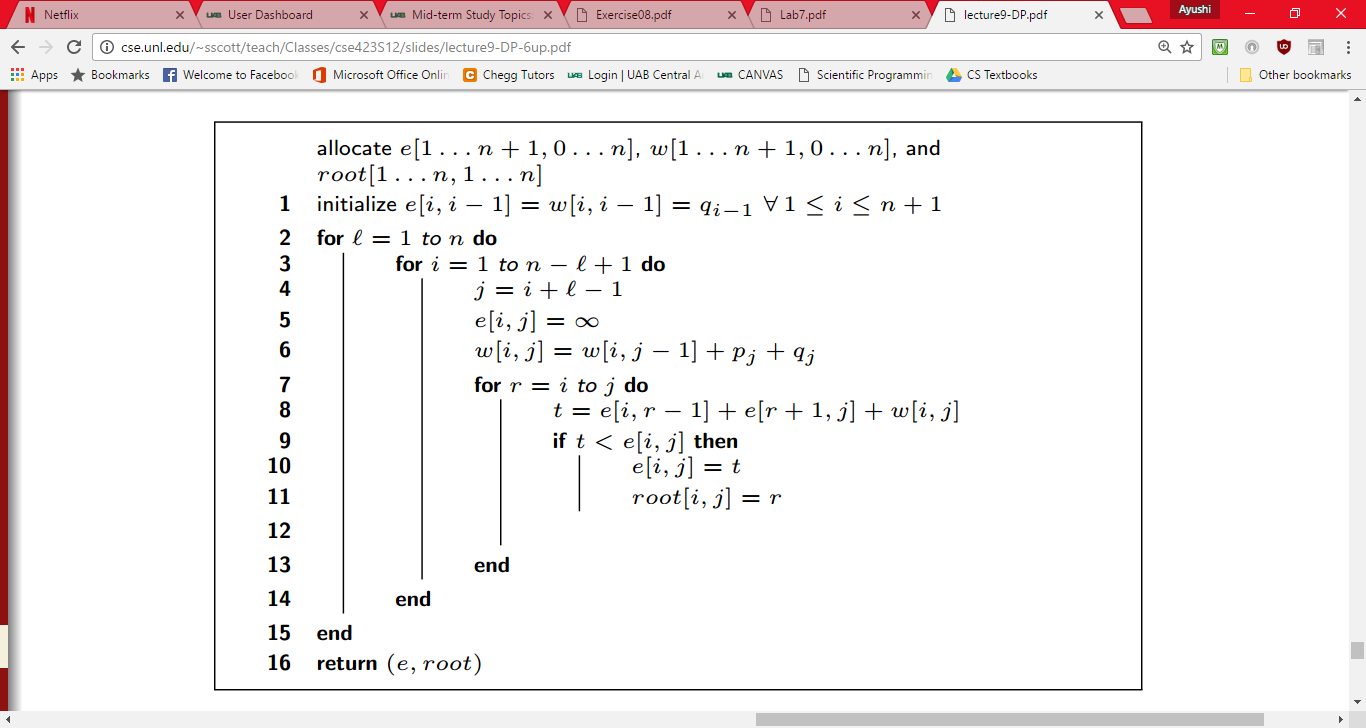
* 1. (3 points) Exercise 15.2-6, page 378.

Show that a full parenthesization of an n-element expression has exactly n-1 pairs of parentheses.

* Given that parentheses group two matrices at a time, there must be exactly n − 1 pairs of parentheses in a full parenthesization of an n-element expression. To see this, consider arbitrarily parenthesizing two matrices in a chain. This effectively replaces the two grouped matrices with one multiplied matrix. This process of grouping two elements can occur n − 1 times before we are left with a single item. ("Chapter Review Answers.")
  1. (3 points) Exercise 15.3-4, page 390.
* Use a counter example to prove the claim gives suboptimal solution. Consider the following case A1 is 5x6, A2 is 6x3, A3 is 3x2. If you use her claim, then the result is (A1A2) A3 = 120. But we can group in another way, (A1(A2A3)) = 96. So, her claim is not correct.

*(Advanced Algorithm Homework 1)*

* 1. (10 points) Exercise 15.5-1, page 403. Hint: consider the algorithm PRINT-OPTIMAL-PARENS on page 377 as a possible source of ideas.



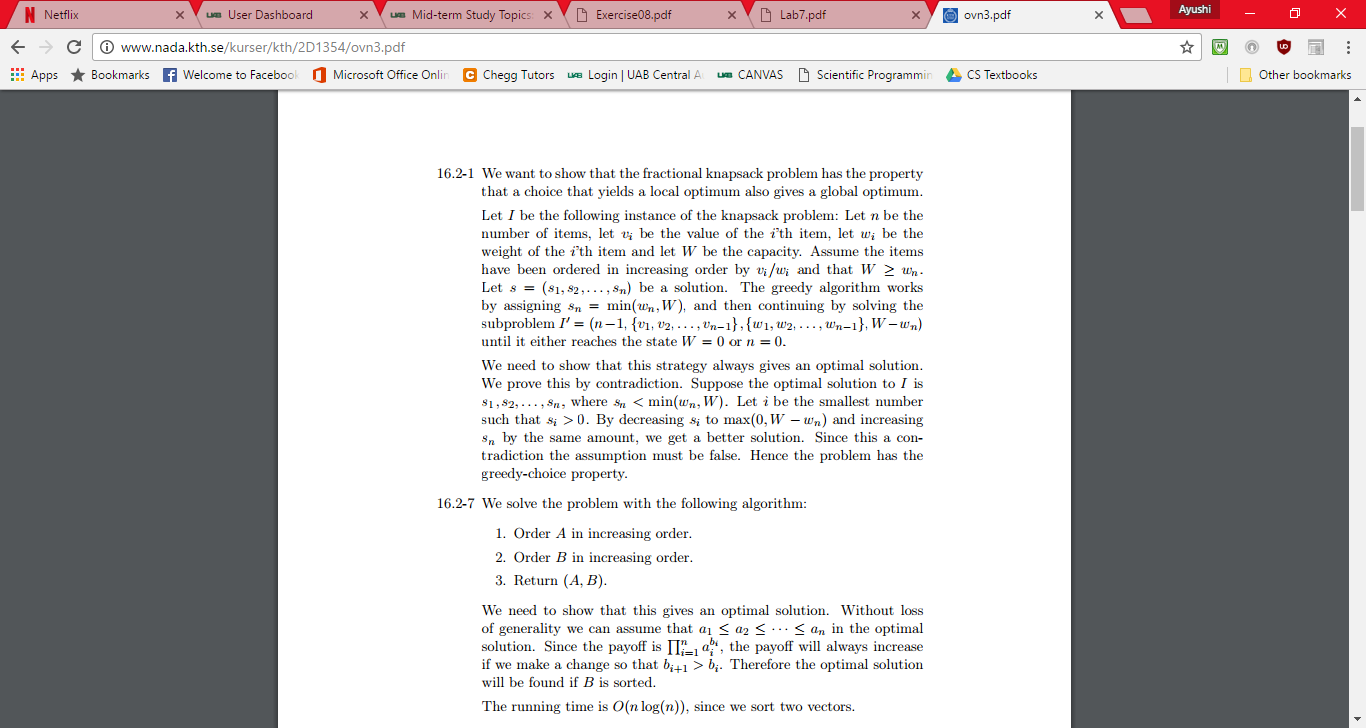
(Scott)

1. Work the following Exercises from Chapter 16 of the text:
   1. (5 points) Exercise 16.1-2, page 422.

* The proposed approach – selecting the last activity to start that is compatible with all previously selected activities – is really the greedy algorithm but starting from the end rather than the beginning. Another way to look at it is as follows. We are given a set S = {a1, a2, · · · , an} of activities, where ai = [si , fi), and we propose to find an optimal solution by selecting the last activity to start that is compatible with all previously selected activities. Instead, let us create a set S’= {a’1, a’2, · · · , a’n}, where a’i = [si , fi). That is, a’i is ai in reverse. Clearly, a subset of { a1, a2, · · · , an} ⊂ S is mutually compatible if and only if the corresponding subset {a’1, a’2, · · · , a’n} ⊂ S’ is 5 also mutually compatible. Thus, an optimal solution for S maps directly to an optimal solution for S’ and vice versa. The proposed approach of selecting the last activity to start that is compatible with all previously selected activities, when run on S, gives the same answer as the greedy algorithm from the text – selecting the first activity to finish that is compatible with all previously selected activities – when run on S’. The solution that the proposed approach finds for S corresponds to the solution that the text’s greedy algorithm finds for S’, and so it is optimal.

(Haung)

* 1. (3 points) Exercise 16.2-1, page 427.



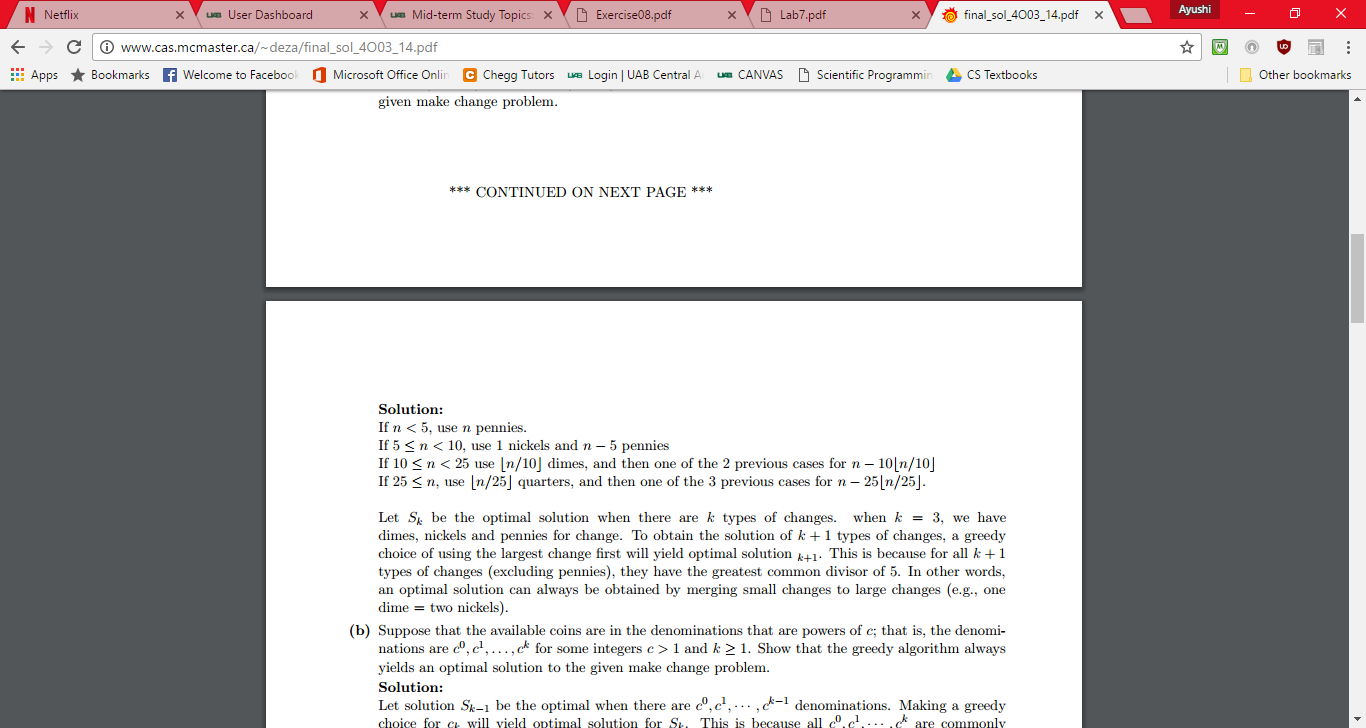
(http://www.nada.kth.se/kurser/kth/2D1354/ovn3.pdf)

* 1. (5 points) Exercise 16.2-4, page 427.
* The optimal strategy is the obvious greedy one. Starting with both bottles full, Professor Gekko should go to the westernmost place that he can refill his bottles within m miles of Grand Forks. Fill up there. Then go to the westernmost refilling location he can get to within m miles of where he filled up, fill up there, and so on. Looked at another way, at each refilling location, Professor Gekko should check whether he can make it to the next refilling location without stopping at this one. If he can, skip this one. If he cannot, then fill up. Professor Gekko doesn’t need to know how much water he has or how far the next refilling location is to implement this approach, since at each fillup, he can determine which is the next location at which he’ll need to stop. This problem has optimal substructure. Suppose there are n possible refilling locations. Consider an optimal solution with s refilling locations and whose first stop is at the kth location. Then the rest of the optimal solution must be an optimal solution to the subproblem of the remaining n − k stations. Otherwise, if there were a better solution to the subproblem, i.e., one with fewer than s−1 stops, we could use it to come up with a solution with fewer than s stops for the full problem, contradicting our supposition of optimality. This problem also has the greedy-choice property. Suppose there are k refilling locations beyond the start that are within m miles of the start. The greedy solution chooses the kth location as its first stop. No station beyond the kth works as a first stop, since Professor Gekko would run out of water first. If a solution chooses a location j < k as its first stop, then Professor Gekko could choose the kth location instead, having at least as much water when he leaves the kth location as if he’d chosen the jth location. Therefore, he would get at least as far without filling up again if he had chosen the kth location. (Haung)
  1. (3 points) Exercise 16.3-3, page 436.

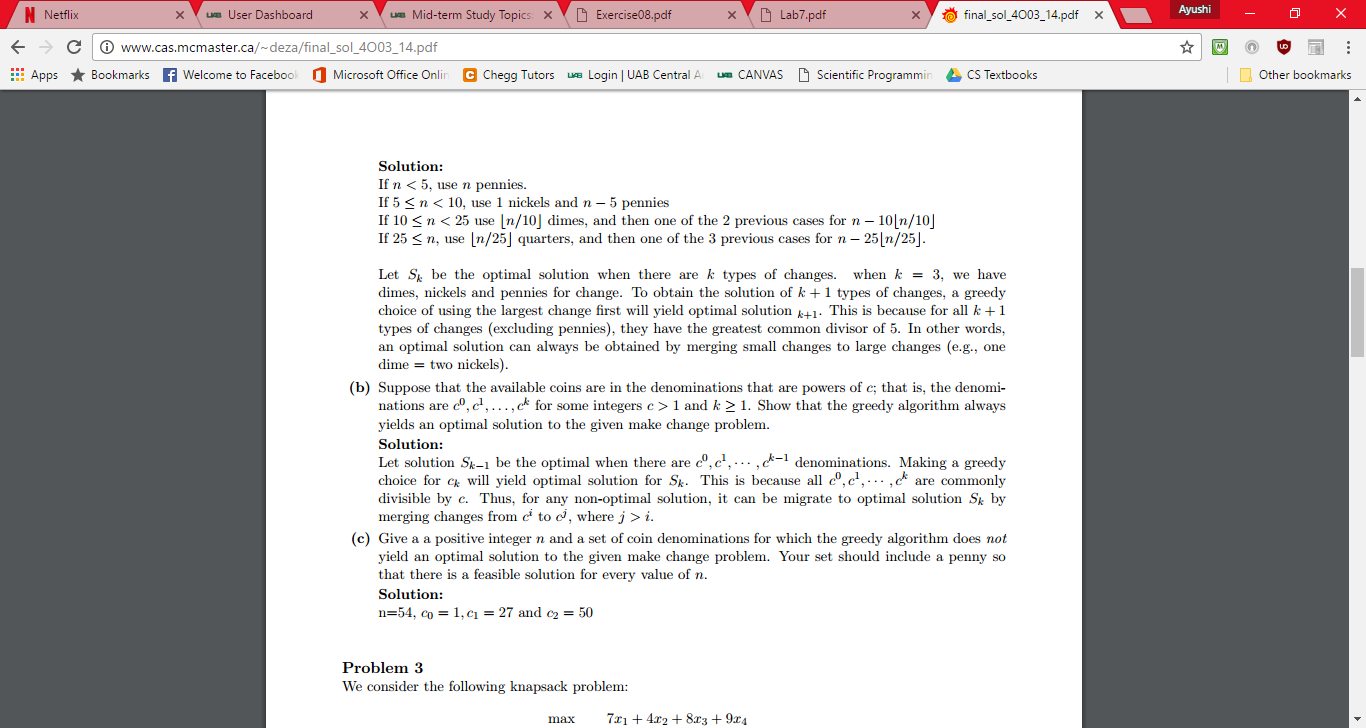
Solve the first question only regarding the first 8 Fibonacci numbers. e. For three points extra credit, extend your solution to Exercise 16.3-3 to n Fibonacci numbers, as outlined in the second question to the exercise.

* The huffman code for these characters are: C(h) = 1, C(g) = 01, C(f) = 001, C(e) = 0001, C(d) = 00001, C(c) = 000001, C(b) = 0000001, C(a) = 0000000. If we switch 0 and 1, we get another code; the codes of a and b can also be exchanged. The generlized case is that the huffman codes are 1, 01, 001, ..., 0 k−11, 0 k for the characters from high frequency to low frequency, assuming we have k + 1 characters in total. The code tree is a linear structure such that each level has exactly one leaf, except the root which has no leaves and the bottom level which has two leaves. The characters with lower frequencies stay at lower level.

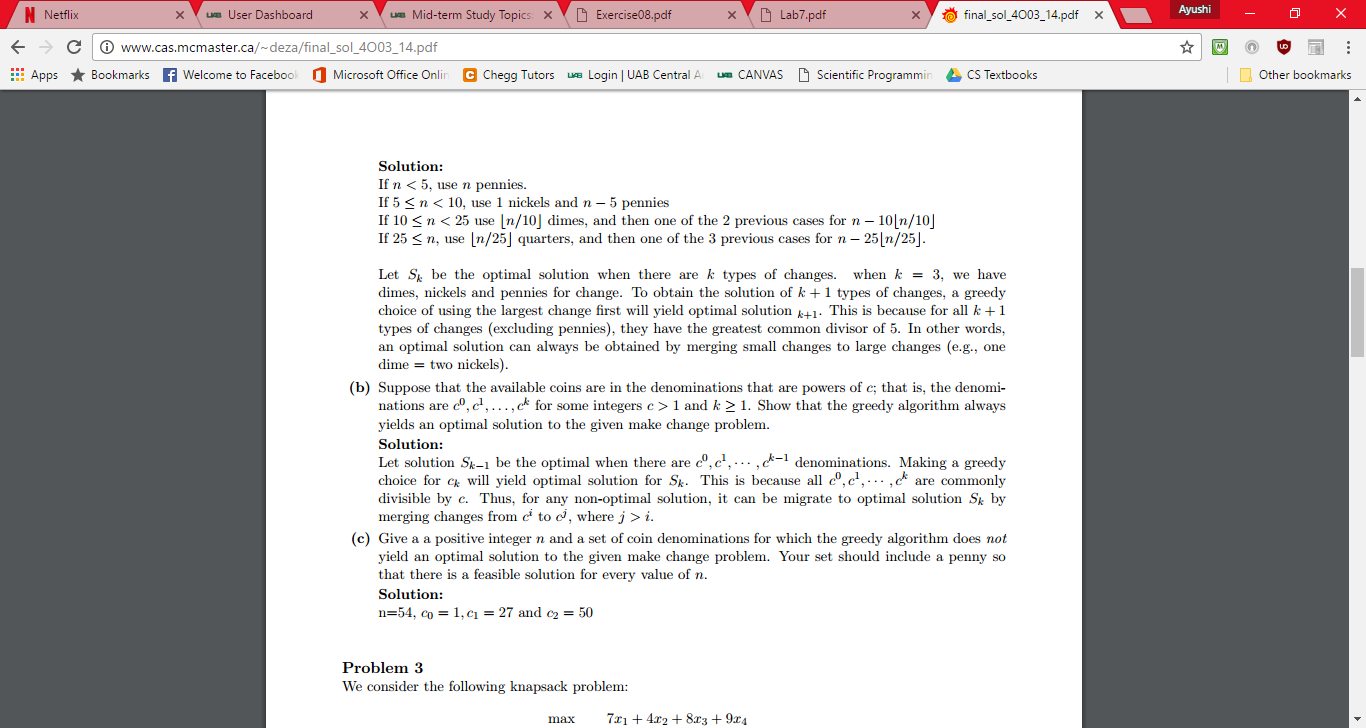
(*Midterm I (22C:231, Spring 2015))*

1. (9 points) Solve Problem 16-1, page 446. The problem has four parts, a, b, c, and d. Parts a, b, and d are required, for 3 points each. Part c is optional and may be completed for 3 points extra credit.
   1. 

(Deza)

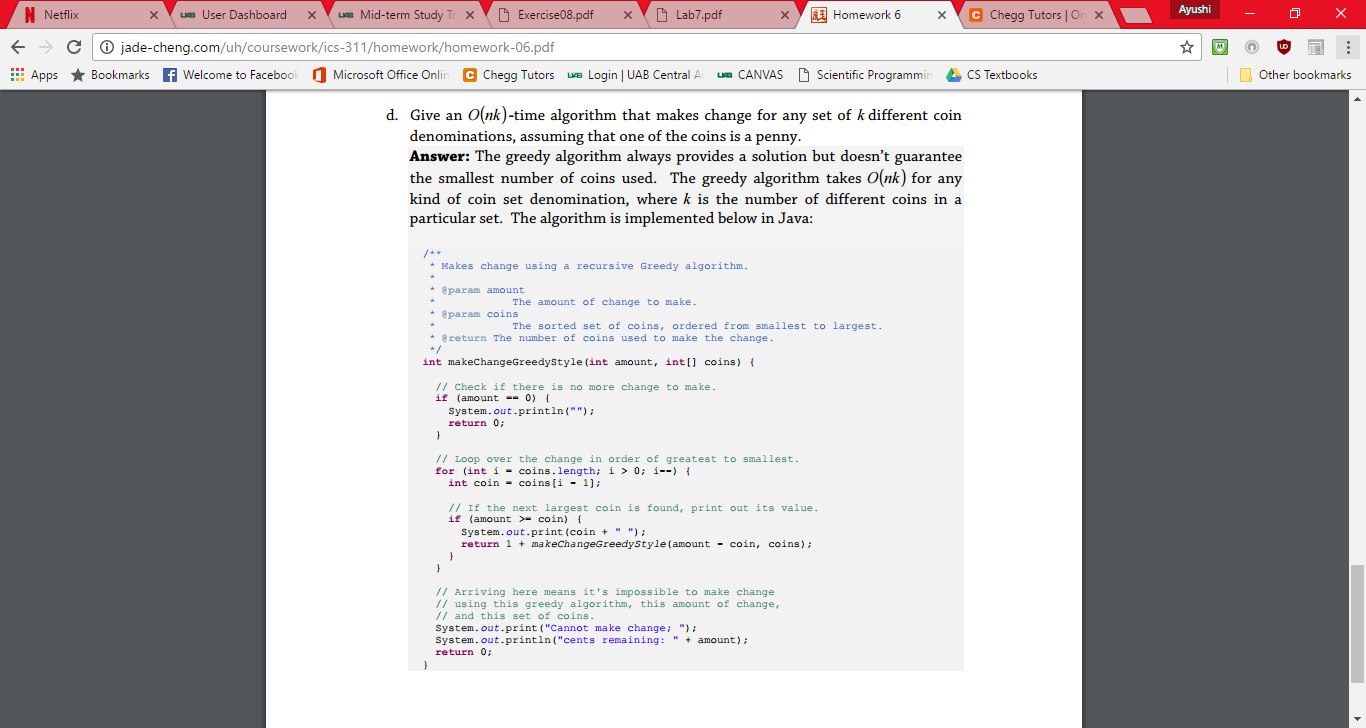


(Deza)





(Deza)

* 1. 

(Cheng)

Works Cited

*Advanced Algorithm Homework 1* (n.d.): n. pag. Web.

"Chapter Review Answers." *Understanding Neural Networks and Fuzzy Logic*(2009): n. pag. Web.

Cheng, Jade Yu. *ICS 311 Homework 6* (n.d.): n. pag. 15 Sept. 2008. Web.

Deza, Antoine. (n.d.): n. pag. *COMP SCI / SFWR ENG 4O03 / 6O03: McMaster University Final Exam Solutions*. 17 Dec. 2013. Web.

Haung. *CSE 5311 Homework 3 Solution* (n.d.): n. pag. Web.

*Midterm I (22C:231, Spring 2015)* (n.d.): 1-3. Web.

Scott, Stephen. "Lecture 09 — Dynamic Programming (Chapter 15)." (n.d.): n. pag. *Computer Science & Engineering 423/823 Design and Analysis of Algorithms*. University of Nebraska. Web.