Arpan Patel
CSC 321 Final Exam

1. (15 points) True or false:

a. The class NP contains some problems that can be solved in polynomial time.

Answer: True

b. All of the problems in NP can be solved in worst-case exponential time.

Asnwer: False

c. There are problems that can be reduced to the Hamiltonian circuit problem and for which we have polynomial-time algorithms.

Answer: False

d. There are problems that the Hamiltonian circuit problem can be reduced to and for which we have polynomial-time algorithms.

Answer: True

2. (15 points) Give an asymptotic bound for each of the following:

a. $T(n) = 3T(n/3) + \lg(n)$

Answer: O(n)

b. $T(n) = 3T(n/6) + n$

Answer: $\Theta(n)$

c. $T(n) = 9T(n/4) + n$

Answer: $\Theta(n^{\log_4 9})$

d. $T(n) = 4T(n/2) + n^2$

Answer: $\Theta(n^2 \log n)$

3. (10 points) Analyze the running time of the piece of code below. Your answer should contain the **summations** that arise from the analysis, the **closed form** of those sums, and the **asymptotic running time**.

```
countPairSum(a, sum) {
  n = a.size
  count = 0
  for i = 1 to n {
    for j = i to n {
```

```
      if a[i] + a[j] == sum {
        count += 1
      }
    }
  }
  return count
}
```

$$\text{Summations: } \sum_{i=1}^{n} * \sum_{j=i}^{n} *(1)$$

Closed form: n*(n+1)/2

Asymptotic Running Time. O(n^2)

4. (10 points) Analyze the running time of the piece of code below, assuming that the array a is sorted and knowing that the running time of binary search is lg(*n*). Your answer should contain the **summation** that arises from the analysis, the **closed form** of that sum, and the **asymptotic running time**.

```
countPairSum(a, sum) {
  n = a.size
  count = 0
  for i = 1 to n {
    difference = a[i] - sum
    if binarySearch(a, difference) {
      count += 1
    }
  }
  return count
}
```

$$\text{Summations: } \sum_{i=1}^{n} * 1 * \log n$$

Closed form: n*log n

Asymptotic Running Time. O(n*log(n))

5. (10 points) Below is the pseudocode for solving the "Find max subarray" problem. Please analyze the running time under the assumption that the method "findMaxCrossingSubarray" runs in time $O(n)$, where n is the length of the array it is given. Your answer should contain the **recurrence relation** and the **asymptotic running time**.

```
findMaxSubarray(a, i, j) {
```

```
  if i == j {
    return (i, j, a[i])
  }
  mid = (i + j) / 2
  (lefti, leftj, leftsum) = findMaxSubarray(a, i, mid)
  (righti, rightj, rightsum) = findMaxSubarray(a, mid+1, j)
  (crossi, crossj, crosssum) = findMaxCrossingSubarray(a, i, mid, j)
  if leftsum > rightsum and leftsum > rightsum {
    return (lefti, leftj, leftsum)
  } else if rightsum > leftsum and rightsum > crosssum {
    return (righti, rightj, rightsum)
  } else {
    return (crossi, crossj, crosssum)
  }
}
```

Recurrence relation: c (n) = 2*c(n/2)+n
Asymptotic running time. O(n*log(n))

6. (15 points) Simulate Prim's MST algorithm: Here is a list of edges in an undirected graph containing 9 vertices. Each edge has the form (v, w, weight), where v and w are the labels of the vertices the edge connects. Your answer should contain the weight of a minimal weight spanning tree found by Prim's algorithm, along with a list of the edges in that spanning tree.

```
(a, b, 4)
(b, c, 8)
(c, d, 7)
(d, e, 9)
(e, f, 10)
(f, g, 2)
(g, h, 1)
(h, a, 8)
(b, h, 11)
(c, i, 2)
(c, f, 4)
(i, g, 6)
(d, f, 14)
(h, i, 7)
```

Answer:
A list of the edges in that spanning tree
(A TO B, 4)
(A TO H, 8)
(H TO G, 1)
(I TO C, 2)
(D TO E, 9)
(D TO C, 7)
(C TO F, 4)
(G TO F, 2)

WEIGHT OF A MINIMAL WEIGHT IS (4+8+1+2+9+7+4+2=37)

Arpan Patel
CSC 321 Final Exam

7. (15 points) Simulate the longest common subsequence algorithm on the two strings below using the algorithm in the book. Your answer should contain the table created by the algorithm, along with a longest common subsequence.

ABCBDAB
BDCABA

	0	A	B	C	B	D	A	B
0	0	0	0	0	0	0	O	0
B	0	0	1	1	1	1	1	1
D	0	0	1	1	1	2	2	2
C	0	0	1	2	2	2	2	2
A	0	1	1	2	2	2	3	3
B	0	1	2	2	3	3	3	4
A	0	1	2	2	3	3	4	4

Longest common subsequence above table is BDAB

8. (20 points) Implement Kruskal's algorithm: This zip file contains incomplete implementations of Python and Java programs for Kruskal's MST algorith. The only thing missing is the implementation of the method that actually carries out Kruskal's algorithm. Select one of these programs and, using the pseudocode in Algorithm 7.2.4 on pp. 279-280, implement that method.

When you run the program, the main program will test your method on two different graphs. The answers you should get are indicated in comments in the source code.

Answer: Please look at the zip. Final inside the zip for the code.

9. (20 points) Implement the dymamic programming coin changing problem: In the language of your choice, implement the algorithm 8.2.1 on pp. 330-1. It should take as input a list of denominations and an amount and then print the least number of coins needed to make change for that amount. You may use these sample inputs to test your program:

- Denominations: 1, 5, 10, 25, 50; amount: 96
- Denominations: 1, 5, 9, 16; amount: 36
- Denominations: 1, 3, 4; amount: 6

Answer: Arpan_dynamic_coin_change.py file for this. In the zip folder.

Submission instructions

Create a document with the answers for questions 1 through 7. Create source files for the implementations in questions 8 and 9. Place all of these into a folder called "final". Zip that folder (that is, zip it so that the "final" folder is in the zip file) into a file called "final.zip". Make sure the file is called exactly that! Don't include your name or anything else in the file name. Submit the final.zip file to the drop box provided.

Arpan Patel
CSC 321 Final Exam

Last updated Wednesday, March 8th, 2017.