Algorithms and Data Structures – Final Exam

Theoretical questions

1. (4 points) Among functions $2n - \log n$, $n^2 + 2\log n^4$, 2^n, $2n\log n$, and n^3 which ones are in $O(n^2)$? Which ones are in $\Omega(n^2)$?

2. (a) (3 points) Claim the Cut-Property Lemma (in the form we used it in class, i.e. when all edge-weights of the graph are different).

   (b) (2 points) Describe how Prim's algorithms works.

   (c) (5 points) Prove that Prim's algorithm is correct based on the correctness of the Cut-Property Lemma. (You don't have to prove the correctness of the Cut-Property Lemma.)

   OR

   (a) (2 points) Give the definition of pseudo-polynomiality. (I.e. what does it mean to have a pseudo-polynomial algorithm?)

   (b) (2 points) Describe the optimization version of the Knapsack Problem.

   (c) (6 points) Describe the pseudo-polynomial algorithm for the Knapsack Problem we designed in class, explain why it is correct, and explain why it is pseudo-polynomial.

3. (6 points) Describe how hashing with open adressing works if we use linear probing: how do we use the hash function, how do we search, insert, and delete.

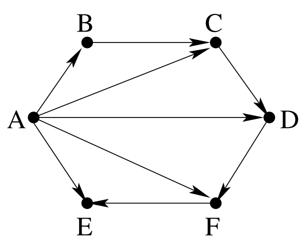
4. (6 points) Let's suppose that $P \neq NP$. For each of the following decision problems decide whether it is in P or not.
   (a) Connectivity of an undirected graph.
   (b) 2-colorability of an undirected graph.
   (c) Given an undirected graph G, we have to decide whether it has a cycle of length 2014.

Problems

5. (4 points) Our task is to store some integers from the set $\{1, 2, 3, \ldots, n\}$. Give a data structure where
(a) we can insert a new number into the data structure,
(b) we can delete a given (stored) number from the data structure, and
(c) we can decide whether a given number is stored in the data structure or not.
All the operations should be performed in $O(1)$ (i. e. constant) time.

6. (5 points) In a red-black tree T, on a path leading from the root to a NULL-leaf the nodes are black, red, black, and black. What is the minimum number of the nodes in T?

7. (5 points) We use Bellman-Ford algorithm for the graph below from source A. Using the first three rows of the array D given below, determine the weights of the edges and complete the remaining rows of the array.



	A	B	C	D	E	F
0.	0	∞	∞	∞	∞	∞
1.	0	5	10	12	15	11
2.	0	5	6	11	13	9
3.						
…						

8. (10 points) We would like to store phone numbers of a big company's clients. Each client has a unique ID but a client can have multiple (but ≤ 100) phone numbers and one phone number can be shared by multiple (but ≤ 100) clients.

Design a data structure if the required operations are the following:
INSERT_PHONE(x,y): inserts a new (client x, phone number y) pair
DELETE_PHONE(x,y): deletes a (client x, phone number y) pair
DELETE_CLIENT(x): deletes the client with ID x
SEARCH_CLIENT(x): returns all phone numbers of client x
SEARCH_PHONE(y): returns all clients with phone number y

All of these operations should be performed in $O(\log n)$ steps where n is the number of clients stored in the data structure. (Since each client can have at most 100 phone numbers, the number of the stored phone numbers is at most $100n$).

9. (10 points) Decide whether the following decision problem is in P or is NP-complete:

Input: an undirected graph G and a subset S of the vertices

Question: Does G has such a spanning tree where the leaves of the tree and the subset S are identical?

Optional problems for extra credit

1. (5 points) We would like to store bit sequences of the form $b_0 b_1 \ldots b_n$ (i.e. the common length of the sequences is $n + 1$). b_0 is a parity bit, i.e. it is 1 if and only if we have an odd number of 1's within the sequence $b_1 b_2 \ldots b_n$. We use linear probing with hash function $h(K) = K \bmod M$ (here we take the bit sequence $b_0 b_1 \ldots b_n$ as an integer, for example 1010 is 6). In which case do we have fewer collisions, when $M = 2^n$ or when $M = 2^n + 1$? (As usual, M denotes the size of the hash table.)

2. (5 points) Let $a_1, a_2, \ldots, a_n$ be a sequence of different integers of length n. A subsequence of this sequence is formed by choosing some members of the original sequence (while preserving their original order): $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$, where $k \geq 1$ and $i_1 < i_2 \ldots < i_k$. (For example $1, 4, 2, 6, 3$ is a subsequence of $9, 1, 4, 5, 8, 2, 10, 6, 12, 3$.)

   We would like to find the longest subsequence of a given sequence of length n where the subsequence's elements are in sorted order (in the sequence above it's $1, 4, 5, 8, 10, 12$).

   Design an algorithm for this task, the running time should be $O(n^2)$.