NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING

EXAMINATION FOR
Semester 1, AY2013/2014

CS3230 – DESIGN AND ANALYSIS OF ALGORITHMS

Nov/Dec 2013 Time Allowed: 2 hours

Instructions to Candidates:

1. This examination paper consists of **FOUR** questions and comprises **FIFTEEN (15)** printed pages, including this page.
2. Answer **ALL** questions.
3. Write ALL your answers in this examination book.
4. You can bring your **TEXT BOOK** and/or **LECTURE NOTES** in the examination.

Matric. Number: ____________________

QUESTION	POSSIBLE	SCORE
Q1	20	
Q2	20	
Q3	20	
Q4	20	
TOTAL	**80**	

IMPORTANT NOTE:

- *You can **freely quote** standard algorithms and data structures covered in the lectures and homeworks. Explain **any modifications** you make to them.*

Q1. (20 points) Short questions

Please answer parts (a)-(d).

(a) (4 points)
What is the worst case and average case running time for (1) sorting n integers using quick sort and (2) inserting n integers into a binary search tree?

Your answer:

	(1) sorting	(2) BST
Worst case running time:	______________	______________
Average case running time:	______________	______________

(b) (6 points)
Suppose that you are given an algorithm S that can find the rank-n integer of an integer array A[1..2n] in O(n) time. By utilizing the algorithm S, give a linear time algorithm to compute the rank-n integer in B[1..4n].

Your answer:

Q1. (continued...)

(c) (4 points)
Rank the following functions in ascending order of growth?
$f_1(n) = n^{1.5}$, $f_2(n) = \frac{1}{n}\sum_{i=1}^{n} i^2$ and $f_3(n) = \sum_{i=1}^{n} i^{1.5}$.

Your answer:

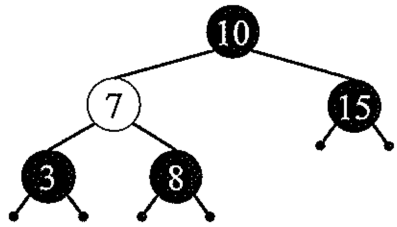
(d) (6 points)
Suppose we perform a sequence of n operations in which the i^{th} operation costs 4i if i is an exact power of 2, and 1 otherwise. Determine the amortized cost per operation. You can use any method you like. You need to justify your answer.

Your answer:

Q2. (20 points) Red-black tree and Graph

Please answer parts (a)-(b).

(a) (5 points)
Give a sequence of insertion so that we can obtain the following red-black tree. (Note 1: white color node represents red color. Note 2: you need to use the method stated in the lecture node or the method in the [CLRS] book.)



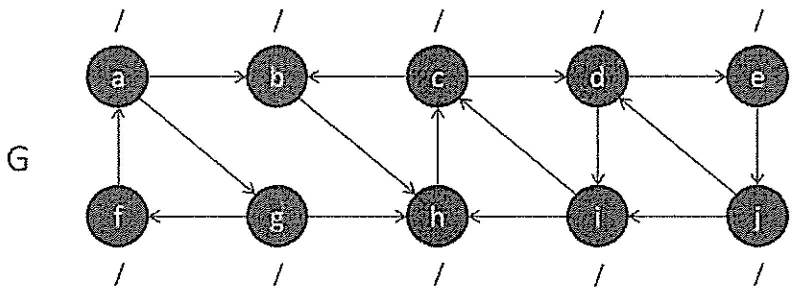
Your answer:

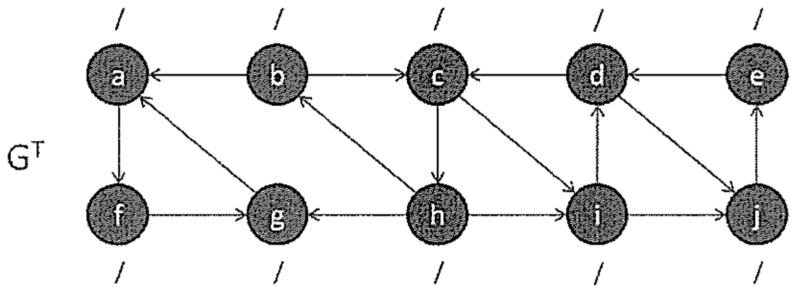
Q2. (continued...)

(b) (15 points)
Consider the following directed graph G and its transpose G^T. Find all strongly connected components of G by executing the strongly connected component algorithm. Please answer the following questions.

(i) **(6 points)** For each vertex in G and G^T, please mark the discovery time and the finishing time when we execute the strongly connected component algorithm. Please also draw the edges of the DFS tree in both G and G^T. (When you perform DFS in G, please assume you will visit the vertices in alphabetical order.)

Your answer:





Q2. (continued...)

(ii) **(4 points)** From the result in (i), write down the strongly connected components.

Your answer:

(iii) **(5 points)** In G, write down the edge types of (g, h), (a, b), (c, b), (d, i) and (b, h). (Note that there are 4 edge types: tree edge, back edge, forward edge and cross edge.)

Your answer:

(g, h): ______________________________

(a, b): ______________________________

(c, b): ______________________________

(d, i): ______________________________

(b, h): ______________________________

Q3. (20 points) Greedy and Dynamic Programming

Subset Sum Problem. Let A[1..n] be a set of positive integers (sorted in increasing order), and s is a positive integer. Is there any subset *B* of *A* such that the sum of the elements in *B* equals to s?
For example, A = (1, 2, 5, 9, 20), s=22. Then, B = (2, 20).

(a) **(15 points)** A super-increasing sequence *A* is a sequence such that the next term of the sequence is greater than the sum of all preceding terms. In other words,

$A_{k+1} > \sum_{j=1}^{k} A_j$, where A_j is the **jth** term of **A**.

For example, A = (1, 2, 5, 9, 20) is a super-increasing sequence, because 2>1; 5>1+2; 9>1+2+5; and 20>1+2+5+9.

i. **(3 points)** For A=(1, 2, 5, 9, 20), find the corresponding set B for the integer **s=32, s=25, s=23.**

Your answer:

s=32: ______________________________

s=25: ______________________________

s=23: ______________________________

Q3. (continued...)

ii. **(4 points)** Please propose a greedy algorithm which solves the subset sum problem when **A is super-increasing.**

Your answer:

iii. **(2 points)** What is the running time of your algorithm?

Your answer:

Q3. (continued...)

iv. **(6 points)** Prove that your proposed algorithm is correct.

Your answer:

Q3. (continued...)

(b) **(5 points)** Professor Perfect thinks that dynamic programming can be used to solve the problem when A is **not super-increasing.**
Please provide a recursive relation for the dynamic programming algorithm which solves the subset sum for any **A. Justify the correctness** of the recursive relation proposed.

Your answer:

Q4. (20 points) NP-hardness and approximation

Knapsack problem: Consider a set of n items where the i^{th} item is of weight w_i and of value v_i. Suppose the maximum load of your car is W and $w_i \leq W$ for all i. We aim to find a subset of items $S \subseteq \{1, \ldots, n\}$ such that the value $\sum_{i \in S} v_i$ is maximized while $\sum_{i \in S} w_i \leq W$.

The decision version of the problem asks if its total value $\sum_{i \in S} v_i \geq K$.

Please answer parts (a)-(c).

(a) (5 points) Show that Knapsack problem is in NP.

Your answer:

Q4. (continued...)

(b) (7 points) Prove that there exists a polynomial-time reduction from the subset-sum problem to the Knapsack problem. Describe your reduction clearly.

Subset-sum problem: Given a set A of positive integers and an integer s, the subset-sum problem decides if there exists a subset B of A whose sum is s.)

Suppose the subset-sum instance is A={1, 2, 5, 9, 20} and s=22. Give the Knapsack problem instance of your reduction.

Your answer:

(This page is blank. Use it for your answer.)

Q4. (continued...)

(c) (8 points) Consider the following algorithm for the Knapsack problem.

1. Sort the items such that $v_i/w_i \geq v_{i+1}/w_{i+1}$ for i=1, ..., n-1.
2. Find the maximum k such that $w_1+\ldots+w_k \leq W$.
3. If $(v_1+\ldots+v_k) \geq v_{k+1}$, then reports {1, ..., k} ; otherwise, reports {k+1}.

(i) **(4 points)** Let OPT be the optimal solution to the Knapsack problem. Let k be the value we found in step 2 of the above algorithm. Show that OPT$\leq$ $v_1+\ldots+v_k+v_{k+1}$.

Your answer:

Q4. (continued...)

(ii) **(4 points)** Show that the approximation ratio of the above algorithm is 2.

Your answer:

-- End of Paper ---