Attempt 1

Question 1.

The master method depends on the following theorem.

Theorem 4.1 (Master theorem)
Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = aT(n/b) + f(n),$$

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$. ■

T(n) = 3T(n/3) +$\theta(n)$. Then, T(n) = $\theta(nlogn)$.

True.

Question 2.

In a red-black tree, the longest path is at most twice the shortest path.

True.

Question 3.

When n is not a power of 2, Strassen's algorithm on two n x n matrices runs in time O(n^{log7}).

False.

Question 4.

A queue can be implemented using three stacks.

True.

Question 5.

A sorted array in descending order is a max d-ary heap.

True.

Question 6.

Inserting a red node somewhere in a red-black tree that maintains the binary-search-tree property but may violate the red-black tree properties, it is possible to restore it to a red-black tree.

True.

Attempt 2

Question 1.

It is possible that searching for an item in a BST of n nodes may require n comparisons.

True.

Question 2.

The running space of dynamic programming is at least in the order of the product of the number of nodes and the number of edges in the underlying subproblem digraph.

False.

Question 3.

Quick sort cannot be made stable.

False.

Question 4.

The idea of dynamic tables is to double the size of the current table to a new table when the load factor is high and reduce the size of the current table by half if the load factor is low.

True.

Question 5.

A sorted array in descending order is a max heap.

True.

Question 6.

Consider a modification to the activity-selection problem in which each activity aihas, in addition to a start and finish time, a value vi. The objective is no longer to maximize the number of activities scheduled, but instead to maximize the total value of the activities scheduled. That is, we wish to choose a set A of compatible activities such that adding them up is maximized. There is no polynomial-time algorithm for solving this problem.

False.

Attempt 3

Question 1.

The running space of dynamic programming is at least in the order of the product of the number of nodes and the number of edges in the underlying subproblem digraph.

False.

Question 2.

Let $T(n) = 3T(n/4) + n \log n$. Then $T(n) = \Theta(n^{3/4} \log n)$.

False.

Question 3.

An algorithm that runs in Θ(t(n)) time means that on input of size n, its worst-case running time does not exceed ct(n) and must be at least c't(n) for some positive constants c and c'.

False.

Question 4.

If m is a matrix. Then list(zip(*m)) returns its transpose.

True.

Question 5.

A common approach of designing a greedy algorithm is to come up with a greedy choice and show that a greedy choice can always appear in an optimal solution.

True.

Question 6.

A randomized algorithm is an algorithm whose analysis depends on the distribution of inputs.

False.

Attempt 4

Question 1.

Given m sorted arrays of equal size n, it is possible to merge them into one sorted array in O(mn log m) time.

True.

Question 2.

Radix sort must start from the most significant digit, then the next significant digit and so on until the least significant digit.

False.

Question 3.

Consider a modification to the activity-selection problem in which each activity aihas, in addition to a start and finish time, a value v_i. The objective is no longer to maximize the number of activities scheduled, but instead to maximize the total value of the activities scheduled. That is, we wish to choose a set A of compatible activities such that adding them up is maximized. There is no polynomial-time algorithm for solving this problem.

False.

Question 4.

It is possible to allocate and deallocate storage for elements within the hash table itself by linking all unused slots into a free list, which is a list of unoccupied slots. Assume that one slot can store a flag and either one element plus a pointer or two pointers. All dictionary and free-list operations can run in O(1) expected time with the free list being singly linked.

False.

Question 5.

The maximum number of elements in a heap of height h is $2^{h+1} - 1$.

True.

Question 6.

When n is not a power of 2, Strassen's algorithm on two n x n matrices runs in time $O(n^{\log 7})$.

False.

Attempt 5

Question 1.

A heap of n keys has height h = floor of log n.

True.

Question 2.

Let Y be the sequence of sorted X in increasing order. Finding the longest common subsequence of X and Y yields the longest monotonically increasing subsequence of X.

True

Question 3.

A randomized algorithm is a deterministic algorithm using a PRNG to complete a task.

True.

Question 4.

The minimum number of elements in a heap of height h is 2^{h+1} - 1.

False.

Question 5.

Let T(n) = 2T(n/2) + n log n. Then the master theorem doesn't apply.

True.

Question 6.

Dynamic programming is about finding a recurrence relation and using memorization to avoid recomputing the same subproblem in solving the recurrence relation.

True.

Attempt 6

Question 1.

It is not possible to sort n numbers in O(n) time using comparisons.

True.

Question 2.

For the same input, bulidng an AVL tree requires less space than building a red-black tree.

False.

Question 3.

Strassen's algorithm on two n x n matrices runs in time $O(n^{\log 7})$ when n is a power of 2. It follows from $\log 7 < 2.81 < 3$ that $O(n^{\log 7})$ is asymptotically faster than the naive matrix multiplications with running time $\Omega(n^3)$. Now you have two square matrices of 512 x 512 dimensions, and you'd need to compute the product of them. Strassen's algorithm obviously should be used.

False.

Question 4.

Insertion sort cannot be made stable.

False.

Question 5.

Quick sort cannot be made stable.

False.

Question 6.

Inserting a red node somewhere in a red-black tree that maintains the binary-search-tree property but may violate the red-black tree properties, it is possible to restore it to a red-black tree.

True.

Attempt 7

Question 1.

If we modify the definition a red-black tree to allow a red node to have a black child but not a black grandchild, with everything other property remaining the same. Then the length of the longest path is at most twice of that of the shortest path.

False.

Question 2.

Define a class for a data structure and the associated operations, the user-defined function __init__(self, v) is used to specify an instance with v being the initial value.

True.

Question 3.

An algorithm that runs in $\Theta(t(n))$ time means that on input of size n, its worst-case running time does not exceed ct(n) and its best-case running time must be at least c't(n) for some positive constants c and c'.

True.

Question 4.

Suppose we perform a sequence of stack operations on a stack whose size never exceeds k. After k operations, we make a copy of the entire stack for backup purposes. The cost of n stack operations, including copying the stack is O(n).

True.

Question 5.

Let $T(n) = T(2n/3) + 1$. Then $T(n) = \Theta(1)$.

True.

Question 6.

Let $T(n) = 2T(n/2) + n \log n$. Then the master theorem doesn't apply.

True.

Attempt 8

Question 1.

For any hash function h, if two items do not collide under h, then their keys must be different.

True.

Question 2.

Suppose you have 10 symbols a1, a2, ..., a10 in a file such that the frequency of ai is the i-th Fibonacci number. Then the Huffman code of the a5 has a length of 4.

False.

Question 3.

The total cost of a tree for prefix codes of symbols in a set is the sum over all internal nodes of the combined frequencies of the two children.

True.

Question 4.

By maintaining an attributed T.bh at the root of a red-black tree T for the black height, it is possible to compute each node's black height on a given path from the root to a leaf in O(1) time when the node is visited.

True.

Question 5.

If m is a matrix. Then list(zip(m)) returns its transpose.

False.

Question 6.

In dynamic tables, when copying elements from an old table to a new table, it should follow the same hashing strategy of the old table but with a new table size.

True.

Attempt 9

Question 1.

A randomized algorithm is a deterministic algorithm that uses a PRNG to make statistically random choices.

True.

Question 2.

Open addressing can always resolve collision without the need of chaining using any probing function so that a table of size n holds exactly n items.

False.

Question 3.

Delete an item from a hash table with open addressing, it suffices to set the bucket occupied by the item to NIL.

False.

Question 4.

A sorted array in ascending order is a min heap.

True.

Question 5.

An algorithm that runs in Ω(t(n)) time means that on input of size n, its worst-case running time must be at least ct(n) for some positive constant c.

False.

Question 6.

Let T(n) = 9T(n/3) + n. Then T(n) = Θ(n^2 log n).

False.

Attempt 10

Question 1.

Suppose that algorithm A runs in time 10^(-20) * 2^(2*n)* + 10^100, then we can say that it runs in O(2^(*n)*)time.

False.

Question 2.

Bucket sort can use any hash function to hash an input number to a bucket.

False.

Question 3.

Selecting the activity of the compatible activity with the earliest start time does not guarantee a selection of maximum number activities.

True.

Question 4.

Probabilistic analysis is to obtain an estimate (e.g., expected value) of the outcome of an algorithm under the assumption that distribution of the input to the algorithm is known.

True.

Question 5.

The minimum number of elements in a heap of height h is 2h.

True.

Attempt 11

Question 1.

For the same input, building an AVL tree is better for searching than building a red-black tree.

True.

Question 2.

Selecting the activity of the compatible activity with the earliest start time does not guantee a selection of maximum number activities.

True.

Question 3.

Suppose that instead of always selecting the first activity to finish as shown in the textbook, we instead select the last activity to start that is compatible with all previously selected activities. This greedy strategy yields an optimal solution.

True.

Question 4.

Open addressing can always resolve collision without the need of chaining using any probing function so that a table of size n holds exactly n items.

False.

Question 5.

Insertion sort cannot be made stable.

False.

Question 6.

Delete an item from a hash table with open addressing, it suffices to set the bucket occupied by the item to NIL.

False.

Attempt 12

Question 1.

Bucket sort can use any hash function to hash an input number to a bucket.

False.

Question 2.

If m is a matrix. Then list(zip(m)) returns its transpose.

False.

Question 3.

It is possible that searching for an item in a BST of n nodes may require n comparisons.

True.

Question 4.

It is possible to sort n numbers in O(n) time using comparisons.

False.

Question 5.

For any hash function h, if two items have different keys, then they don't collide under h.

False.

Question 6.

A heap of n keys has height h = floor of log n.

True.