Final: Algorithms

Part 1: Show your knowledge of algorithms / data structures covered in the course

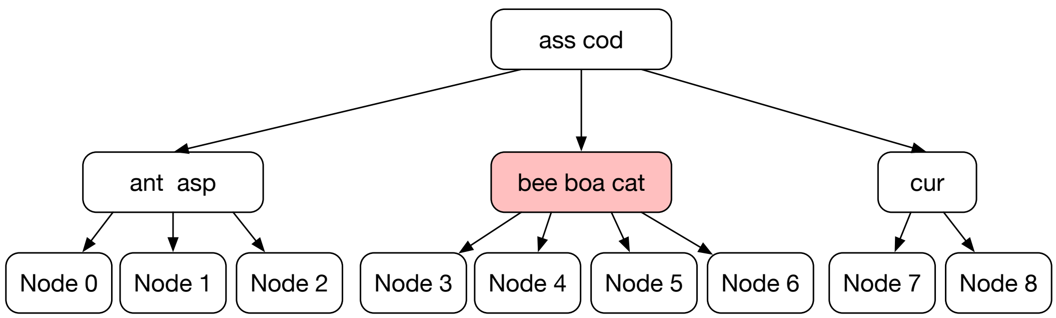
You should solve problems worth 50 points in this section

10 pts **Problem 1:**

An LH structure has exactly 100 buckets. Which bucket houses the record with ID 12345 which is 0x3039 in hexadecimals. (Hint: you can use Python IDLE in order to calculate residues).

10 pts **Problem 2:**

During the insert into a B-tree with maximum number of children per node of three, a split has lead to the following situation:



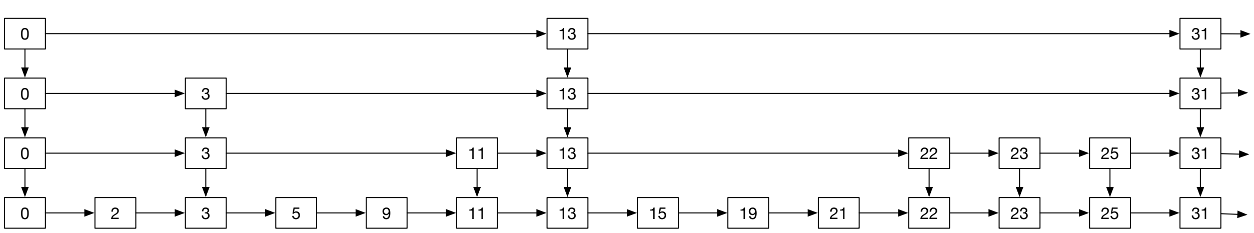
As you can see, there is an overflowing node with three keys and four children. Using a rotation, how is this situation resolved? Show the B-tree after the rotation.

10 pts **Problem 3:**

What is a best encoding for strings with characters 'A', 'E', 'I', 'O', 'U', if the frequencies are 55% (for 'A'), 25% (for 'E'), 15% (for 'I'), 4% (for 'O'), and 1% (for 'U') respectively.

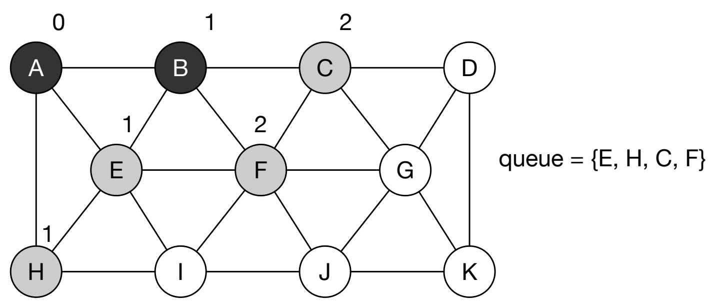
10 pts **Problem 4:**

In the following skip list the value 24 is inserted as a node with three levels. For the top-most level, what are the predecessor and successor. For the medium-level, what are the predecessor and successor. For the lowest level, what are predecessor and successor. (Naming the key is sufficient).



10 pts **Problem 5:**

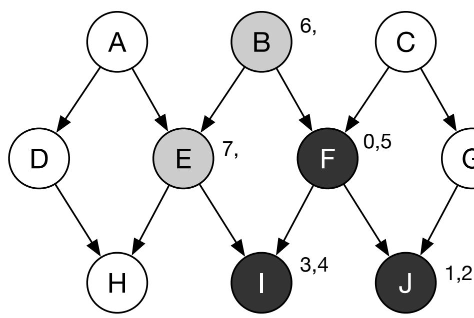
In an execution of BFS starting with node A, the following situation is reached:



The numbers are the distance from A. What is the next step?

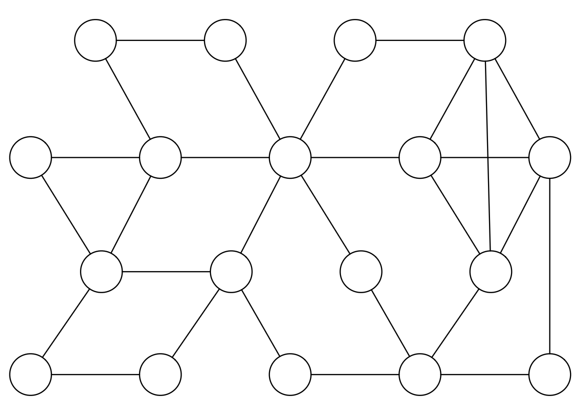
10 pts **Problem 6:**

In an execution of DFS, the following situation has been reached. (The nodes are colored according to the algorithm and adorned with a pair of numbers, discovery and termination time). The algorithm has started in node F with DFS-visit and after visiting I and J has returned. The algorithm then started again with DFS-visit in node B and is currently in E.



10 pts **Problem 7:**

Does the following graph have a Eulerian tour?



Part 2: Show your capability to invent, analyze, and compare algorithms

Work on problems that are worth 50 points.

15 pts **Problem 8:**

On average, $\frac{2 \cdot 2}{6} + \frac{3 \cdot 4}{6} = \frac{16}{6} = \frac{8}{3}$ are needed to find the minimum and maximum of three elements. If the three elements are $\{a, b, c\}$, we compare first a with b and then b with c. If $a < b < c$ or if $a > b > c$ then the maxima and minima are determined with two comparisons. Otherwise, we will need three comparisons. As there are $3!$ ways to order three elements, we obtain this average number of comparisons.

Assume that we want to find the maximum and the minimum of n elements. Assume that n is divisible by three. We divide the array into groups of three. For each group, we determine the maximum and minimum. We then determine the maximum from the group maxima and the minimum from the group minima. How many comparisons are needed in total. How does this compare to the naïve algorithm with $2n - 2$ comparisons. How does this compare to the best algorithm from the book?

15 pts **Problem 9:**

Show that Rudrata's problem is in $\mathcal{NP}$. Rudrata asks whether it is possible to visit all squares on an $n \times n$ chess board exactly once using knight's moves.

15 pts **Problem 10:**

The fraudulent cab driver asks whether it is possible to find a path of length at least l in a given graph and two given nodes A and B. Show that the fraudulent cab driver problem is in $\mathcal{NP}$.

15 pts **Problem 11:**

Give a dynamic programming solution to finding the number of ways in which a number n can be written as a sum of ones, threes, and fours. For instance, if $n = 9$, we can write $9 = 3 + 3 + 3$, $9 = 1 + 1 + 3 + 1 + 3$, $9 = 4 + 1 + 3 + 1$, $9 = 3 + 1 + 1 + 3 + 1$, giving us already four different ways of writing 9 as a sum of ones, threes, and fours.

15 pts **Problem 12:**

Design a divide and conquer problem for calculating k^n (with $k > 0$, $n \geq 0$),

15 pts **Problem 13:**

Give a recurrence for the runtime of the following bogus algorithm.

```
def bogus(n):
   m = n//3
   return bogus(m)+bogus(2*m) + n
```

Can you solve this recurrence using the master theorem?

20 pts **Problem 14:**

Suppose that you are choosing between the following three algorithms:
(A) The algorithm solves the problem by dividing them into five subproblems half the size, recursively solve each subproblem, and then combining the solutions in linear time.
(B) The algorithm solves the problem of size n by recursively solving two subproblems of size $n-1$ and then combining the solutions in constant time.
(C) The algorithm solves the problem of size n by dividing them into nine subproblems of size $n/3$, recursively solving each subproblems, and then combining the solution in $\Theta(n^2)$ time.
What are the running times of these algorithms and which would you pick?

20 pts **Problem 15:**

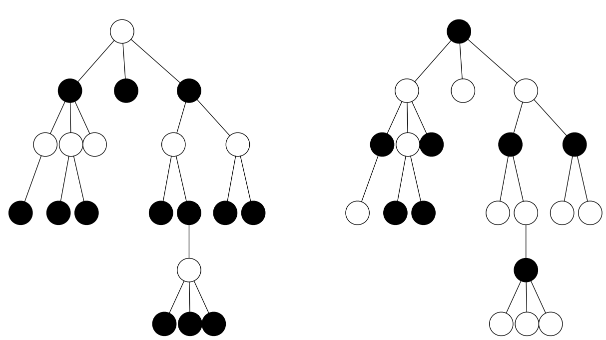
You are given an array with n elements containing duplicates. How can you remove the duplicates in time $O(n \log(n))$?

20 pts **Problem 16:**

Explain why squaring a number and multiplying a number have the same asymptotic run-times.

Problem 17:

30 pts Use dynamic programming to color as many nodes black in a tree if two adjacent nodes cannot be black.



A good and a better way to color tree nodes black.

35 pts **Problem 18:**

The square of a directed graph (V, E) has the same set of nodes V, but there is an edge between $x \in V$ and $z \in V$ exactly if there is another node $y \in V$ such that $(x, y) \in E$ and $(y, z) \in E$. Describe an efficient algorithm to calculate the square of a directed graph. Both of them are given as a list of adjacency matrices. Give the asymptotic run-time of your algorithm.

35 pts **Problem 19:**

Show that the following is **not** true: If in a directed graph there is a path from node $u \in V$ to node $w \in V$ then for any depth first search: $w.d < u.f$.

50 pts **Problem 20:**

Let T be a minimum spanning tree in a graph G. Pick one edge of T and lower its weight by 1. Show that T is still a minimum spanning tree in the graph G with this one weight changed.