# COMP 3804 — Winter 2026 — Problem Set 4

Some useful facts:

1. $1 + 2 + 3 + \cdots + n = n(n+1)/2$.
2. for any real number $x > 0$, $x = 2^{\log x}$.
3. For any real number $x \neq 1$ and any integer $k \geq 1$,
$$1 + x + x^2 + \cdots + x^{k-1} = \frac{x^k - 1}{x - 1}.$$
4. For any real number $0 < \alpha < 1$,
$$\sum_{i=0}^{\infty} \alpha^i = \frac{1}{1 - \alpha}.$$

Master Theorem:

1. Let $a \geq 1$, $b > 1$, $d \geq 0$, and
$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ a \cdot T(n/b) + \Theta(n^d) & \text{if } n \geq 2. \end{cases}$$
2. If $d > \log_b a$, then $T(n) = \Theta(n^d)$.
3. If $d = \log_b a$, then $T(n) = \Theta(n^d \log n)$.
4. If $d < \log_b a$, then $T(n) = \Theta(n^{\log_b a})$.

**Question 1:** Let $G = (V, E)$ be a directed graph, in which each edge $(u, v)$ has a positive weight $wt(u, v)$, let $n = |V|$ and $m = |E|$. This graph is given using adjacency lists: Each vertex $u$ has a list that stores all vertices $v$ such that $(u, v)$ is an edge in $E$. In other words, this list stores all edges *going out* of $u$.

As in class, we denote the length of a shortest path from vertex $u$ to vertex $v$ by $\delta(u, v)$.

Give an algorithm that takes as input this graph $G$, together with two distinct vertices $a$ and $b$. The output of the algorithm is the list of all vertices $v$ such that

1. $\delta(v, a) \leq 20$ (i.e., the shortest path from $v$ to $a$ has length at most 20) *and*
2. $\delta(v, b) \leq 26$ (i.e., the shortest path from $v$ to $b$ has length at most 26).

The running time of your algorithm must be $O((n + m) \log n)$.

Describe your algorithm in plain English (Step 1: Do this. Step 2: Do that. Etc.) You may use any result that was proven in class or in the tutorials. As always, explain why your algorithm is correct and why its running time is $O((n + m) \log n)$.

**Question 2:** Let $G = (V, E)$ be a directed graph, in which each edge $(u, v)$ has a positive weight $wt(u, v)$, and let $s$ be a source vertex in $V$. Let $n = |V|$ and $m = |E|$.

Recall that Dijkstra's algorithm computes for each vertex $v$, the length $\delta(s, v)$ of a shortest path from $s$ to $v$, in total time $O((n + m) \log n)$. The pseudocode for this algorithm is given below.

Assume that each edge weight $wt(u, v)$ is an integer in the set $\{1, 2, \ldots, 2026\}$. Prove that Dijkstra's algorithm can be implemented such that the running time is $O(m + n)$.

*Hint:* A *priority queue* is a data structure that stores any finite sequence of numbers and supports the operations INSERT, EXTRACT_MIN and DECREASE_KEY. An example of a priority queue is a min-heap.

A priority queue is called *monotone* if, during any sequence of operations, the smallest element never decreases. Thus, if at some moment, the smallest element is 45, then later on, the smallest value will always be at least 45. (Note that, even though this was not proven in class, the sequence of operations during Dijkstra's algorithm has this property. You may use this fact.)

Assume that at any moment, any number stored in a monotone priority queue is an integer belonging to the set $\{0, 1, 2, \ldots, k\}$. Start by showing that any sequence consisting of $n$ INSERT-operations, $n$ EXTRACT_MIN-operations and $m$ DECREASE_KEY-operations (these operations may appear in any order) can be processed in total time $O(m + n + k)$.

```
**Algorithm** DIJKSTRA$(G, s)$:
for each $v \in V$
do $d(v) = \infty$
endfor;
$d(s) = 0$;
$S = \emptyset$;
$Q = V$;
while $Q \neq \emptyset$
do $u$ = vertex in $Q$ for which $d(u)$ is minimum;
    delete $u$ from $Q$;
    insert $u$ into $S$;
    for each edge $(u, v)$
    do if $d(u) + wt(u, v) < d(v)$
        then $d(v) = d(u) + wt(u, v)$
        endif
    endfor
endwhile
```

**Question 3:** Let $G = (V, E)$ be a connected undirected graph, in which each edge has a weight. Assume that all edge weights are distinct.

Professor Justin Bieber claims that $G$ has only one minimum spanning tree.

Is Professor Bieber's claim correct? As always, justify your answer.

**Question 4:** Let $G = (V, E)$ be a connected undirected graph, in which each edge has a weight. An edge $\{u, v\}$ is called *annoying* if the graph $G' = (V, E \setminus \{\{u, v\}\})$ is not connected.

Assume there is a unique edge in $E$ with largest weight; denote this edge by $e$.

Prove that $e$ is an edge in every minimum spanning tree of $G$ if and only the edge $e$ is annoying.