Question 1. Minimum Spanning Trees [12 MARKS]

Let $G = (V, E, w)$ be a undirected, connected, weighted graph and assume all weights are distinct (i.e., $w(e) \neq w(f)$ for any pair of edges $e, f \in E$ with $e \neq f$). Suppose $S_0 \subset V$ and $D_0 = \text{cutset}(S_0)$. Suppose $e_0 \in E$ is an edge in D_0.

Given the above assumptions, for each of the questions below circle the most specific correct answer (i.e., if something "always" happens or "never" happens, then "sometimes" will be marked wrong).

Part (a) [2 MARKS]

Suppose e_0 and D_0 are as above, with $|D_0| > 1$, and e_0 is the minimum weight edge in the cutset D_0, then e_0 is

always **sometimes** **never**

in an MST of G.

Part (b) [2 MARKS]

Suppose e_0 and D_0 are as above, with $|D_0| > 2$, and the weight $w(e_0)$ is neither the mininimum nor the maximum of the weights of the edges in the cutset D_0, then e_0 is

always **sometimes** **never**

in an MST of G.

Part (c) [2 MARKS]

Suppose e_0 and D_0 are as above, with $|D_0| > 1$, and e_0 is the maximum weight edge in the cutset D_0, then e_0 is

always **sometimes** **never**

in an MST of G.

Part (d) [2 MARKS]

Suppose e_0 and D_0 are as above, with $|D_0| = 1$, then e_0 is

always **sometimes** **never**

in an MST of G.

Part (e) [2 MARKS]

Suppose e_0 and D_0 are as above, with $|D_0| > 1$, then G

always **sometimes** **never**

has a simple cycle C with two or more edges in D_0.

Part (f) [2 MARKS]

Define $D(v)$ to be the cutset formed from $S(v)$, where $S(v) = \{v\}$ for each $v \in V$, and define

$$B = \{e \mid e \text{ is the minimum weight edge in } D(v) \text{ for some } v \in V\}$$

then the graph (V, B) is

always **sometimes** **never**

acyclic.

Question 2. Divide and Conquer: More Than a Third [18 MARKS]

We are given a list of $n > 0$ objects, say $X = (x_1, x_2, \ldots, x_n)$. The only operation we can do on these objects is equality testing (therefore we can not sort it). We must find all objects which occur in X strictly more than $n/3$ times. (We define $|X|$ to be this n.)

Part (a) [3 MARKS]

Define M to be the maximum number of different objects (e.g., x_i and x_j with $x_i \neq x_j$) that can each appear in X **strictly more** often than $|X|/3$ times. (Note the minimum number of such frequent objects is zero.) Give the value of M and prove (carefully) that it is correct.

Part (b) [12 MARKS]

Finish the pseudo-code of the function `freqThird` below for computing a list, D, of items which occur strictly more than $|X|/3$ times in X. Your algorithm:

- Must (to get any marks) make essential use of the two recursive calls in the code below;
- Can return additional values beside D (but see the next point);
- All returned values must be either constants or lists that all have $O(1)$ length (such as, D, which we showed in (a) is a list of length at most $|M|$);
- Should have a runtime that is as small as possible (part marks are offered here).
- Should include enough comments to assist the marker.

```
// Return the list D of items appearing in X strictly more often than |X|/3 times:
D = freqThird(X, 1, n)

[D, _____] = freqThird(X, a, b)
    // Input: Array X(1..n) of objects, and indicies 1 ≤ a ≤ b ≤ n.
    // Output: The list D described above for the sublist X(k), k = a, ..., b.
    // Describe any additional output here:
    //
    if b > a
        m = floor((b + a - 1) / 2)
        [DL, _____ ] = freqThird(X, a, m)
        [DR, _____ ] = freqThird(X, m+1, b)
        // Note to student: Remember to complete the else clause below.
```

Part (c) [3 MARKS]

Analyze your algorithm's running time.
(For your reference, the Master Theorem states that any function that satisfies a recurrence of the form $T(n) = aT(n/b) + \Theta(n^d)$ has solution $T(n) = \Theta(n^d)$ if $a < b^d$, $T(n) = \Theta(n^d \log n)$ if $a = b^d$, and $T(n) = \Theta(n^{\log_b a})$ if $a > b^d$.)