Problem 1

```
def minimum_water_stops(distances, m):
    n = len(distances)
    water_stops = []
    curr_pos = 0
    while curr_pos < n:
        cand_pos = pos + 1
        d = (distances[cand_pos] - distances[pos])
        while cand_pos < n and m*d <= 2*d:
           cand_pos += 1
        water_stops.append(cand_pos -1)
        curr_pos = cand_pos - 1
    return water_stops
```

We may utilize a contradiction argument to demonstrate the effectiveness of this tactic. Let's say we have an optimal solution where the professor stops at Xi, however, using our greedy approach, the professor would have proceeded to Xj, if he could reach Xj using the same amount of water, that means he could've stopped at Xi, but he didn't under our greedy approach, given that Xj is closer to the final destination, that means that the optimal solution is at best as good as the greedy solution.

Assume that we have an optimal solution that doesn't follow the greedy solution, meaning that we could have station i and followed by station j, where the optimal solution goes to i then j, the greedy solution skips i. If the distance between i and j is proven to be less than 2m, then the greedy solution is correct, that means, the greedy solution offers the local optimal solution, extending the same methodology to the rest of the stations, proving step by step that a greedy solution remains locally optimal, then the overall optimal solution can be at best as good as the overall greedy solution.

Problem 2

Huffman Eq. : $\sum (fi * li) / n$

Given that all characters are roughly equally repeated.

$$fi = n/256$$

li can be determined from the frequency of repetition of the whole set of characters

$$li = log2(256) = 8\, bits$$

substituting in $\sum (fi * li) / n$

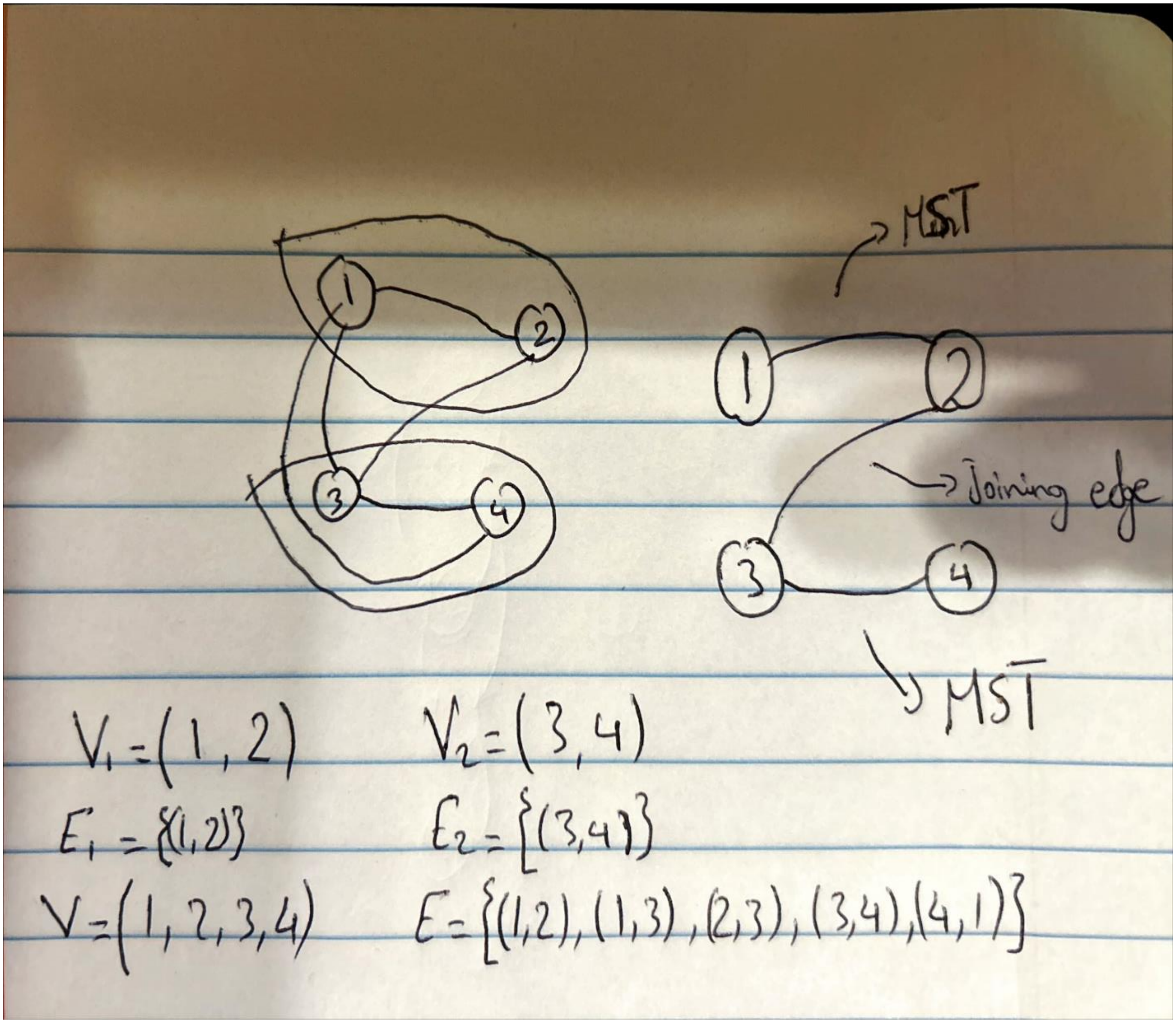
$$\sum (n/256 * 8) / n$$

$$\cong 8bits$$

Problem 3

If (u,v) is contained in a MST, then a crossing cut must separate it into 2 subtrees. If the graphs were joined back to MST using a different edge other than (u,v), then either the new tree is NOT an MST because it now has a more weighted edge, or if its lighter weight, then the original graph was not MST with edge (u,v)

Problem 4



The algorithm should output a minimum spanning tree