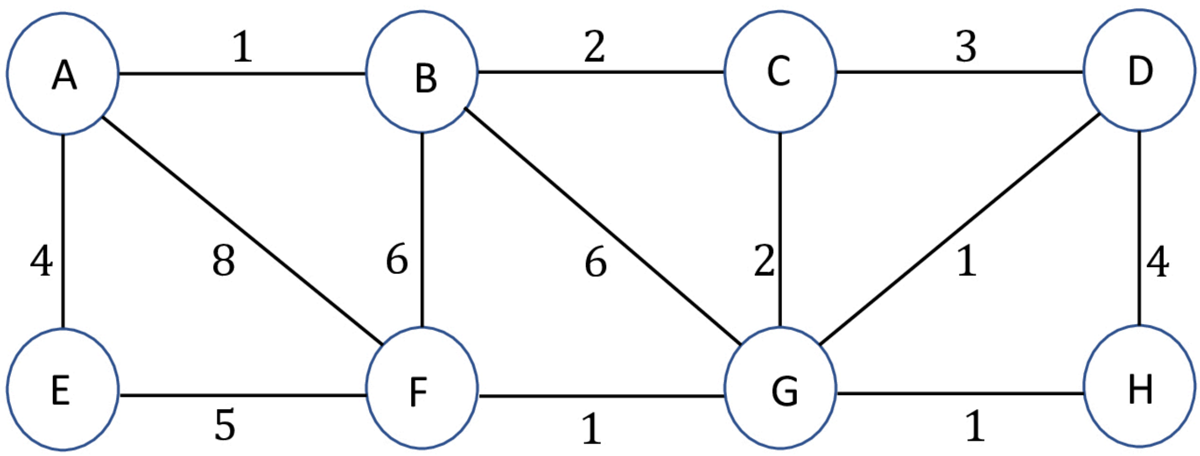
1. (10+10 pts.) Suppose we want to find a minimum spanning tree for the following graph.



Run Prim's Algorithm; whenever there is a choice of nodes, always use alphabetic ordering (eg. start from node A). Mention all intermediate steps, corresponding spanning tree and associated cost.

Run Kruskal's Algorithm; whenever there is a choice of edges, pick an arbitrary one. Mention all intermediate steps, corresponding spanning tree and associated cost.

Solutions.

Prim's Algorithm. Step1: Vertex set: $\{A\}$; Cost: 0

Step2: Vertex set: $\{A, B\}$; Edge Set: $\{AB\}$; Cost: 1

Step3: Vertex set: $\{A, B, C\}$; Edge Set: $\{AB, BC\}$; Cost: 3

Step4: Vertex set: $\{A, B, C, G\}$; Edge Set: $\{AB, BC, CG\}$; Cost: 5

Step5: Vertex set: $\{A, B, C, G, D\}$; Edge Set: $\{AB, BC, CG, GD\}$; Cost: 6

Step6: Vertex set: $\{A, B, C, G, D, F\}$; Edge Set: $\{AB, BC, CG, GD, GF\}$; Cost: 7

Step7: Vertex set: $\{A, B, C, G, D, F, H\}$; Edge Set: $\{AB, BC, CG, GD, GF, GH\}$; Cost: 8

Step8: Vertex set: $\{A, B, C, G, D, F, H, E\}$; Edge Set: $\{AB, BC, CG, GD, GF, GH, AE\}$; Cost: 12

Kruskal's Algorithm. Step1: Vertex set: $\{A, B\}$; Edge Set: $\{AB\}$; Cost: 1

Step2: Vertex set: $\{A, B, F, G\}$; Edge Set: $\{AB, FG\}$; Cost: 2

Step3: Vertex set: $\{A, B, F, G, D\}$; Edge Set: $\{AB, FG, GD\}$; Cost: 3

Step4: Vertex set: $\{A, B, F, G, D, H\}$; Edge Set: $\{AB, FG, GD, GH\}$; Cost: 4

Step5: Vertex set: $\{A, B, F, G, D, H, C\}$; Edge Set: $\{AB, FG, GD, GH, BC\}$; Cost: 6

Step6: Vertex set: $\{A, B, F, G, D, H, C\}$; Edge Set: $\{AB, FG, GD, GH, BC, CG\}$; Cost: 8

Step7: Vertex set: $\{A, B, F, G, D, H, C, E\}$; Edge Set: $\{AB, FG, GD, GH, BC, CG, AE\}$; Cost: 12

2. (7+8 pts.) Under a Huffman encoding of n symbols with frequencies $f_1, f_2, ..., f_n$, what is the longest a codeword could possibly be and why? Give an example set of frequencies that would produce this case, and argue that it is the longest possible.

Solution:

The longest codeword can be of length $n-1$. No codeword can ever be longer than length $n-1$. To see we consider a prefix tree of the code. If a codeword has length n or greater, then the prefix tree should have height n or greater, so it would have at least $n+1$ leaves. Our alphabet is of size n, so the prefix tree can have exactly n leaves and we get a contradiction.

An encoding of n symbols with $n-2$ of them having probabilities $1/2, 1/4, \ldots, 1/2^{n-2}$ and two of them having probability $1/2^{n-1}$ achieves this value. Next, we prove that the depth of the tree is $n-1$, thus the longest code word is of length $n-1$. We use induction for this proof. Specifically, we show that after the k-th iteration, the nodes corresponding to the $k+1$ least frequent symbols are part of a tree with depth $k-1$, where the root node has a frequency of $\frac{1}{2^{n-k-1}}$.

Base Case: In the first iteration, according to the greedy algorithm, we consider the two least frequent symbols, each with a frequency of $\frac{1}{2^{n-1}}$, and place them in a tree. Therefore, after the first iteration, the tree has a depth of 1, and the root node has a frequency $\frac{1}{2^{n-2}}$.

Induction Hypothesis: Assume that after the k-th iteration, the nodes corresponding to the $k+1$ least frequent symbols, namely $\frac{1}{2^{n-k}}, \frac{1}{2^{n-k+1}}, \ldots, \frac{1}{2^{n-1}}, \frac{1}{2^{n-1}}$, are part of a tree with depth $k-1$. The root node of this tree has a frequency of $\frac{1}{2^{n-k-1}}$.

Induction Step: In the $(k+1)$-th iteration, we create a new tree by merging the root node of the existing tree (with frequency $\frac{1}{2^{n-k-1}}$) with the node corresponding to the symbol with frequency $\frac{1}{2^{n-k-1}}$. This results in a new tree that includes the $k+1$ least frequent symbols: $\frac{1}{2^{n-k-1}}, \frac{1}{2^{n-k}}, \ldots, \frac{1}{2^{n-1}}, \frac{1}{2^{n-1}}$. This new tree has a depth of k, and its root node has a frequency of $\frac{1}{2^{n-k}}$.

By induction, the depth of the tree after $n-1$ iterations is $n-1$, as required.