1. Separate Chaining
   1093 % 10 = 3
   1400 % 10 = 0
   3341 % 10 = 1
   7652 % 10 = 2
   4321 % 10 = 1
   5674 % 10 = 4
   8980 % 10 = 0

Index	0	1	2	3	4
Value	1400 8980	3341 4321	7652	1093	5674

Linear Probing

Any repeat of an index will be added 1 to the index that it may land in an empty index column

Index	0	1	2	3	4	5	6
Value	1400	3341	7652	1093	4321	5674	8980

2. We can set up an one way hash functions to report back on the specific key we want to know if it has a object in a set or not. To reduce space we can maybe compress certain sets bit by bit.

3a. The height of the trie is 8 since root counts as 0.
3b. The height is 10 with the extra space and counting the beginning node.
3c. By inserting the longest words first, it will minimize the height. The height will be 9.
3d. By inserting the shortest words first, it will maximize the height. The height will be 11.

4. If prefix free codes aren't used in huffman coding, there will be problem in encoding and decoding. Prefix free code means no 2 code words can be the same. This makes encoding orderly. Without it there will be chaos.
This also allows decoding to process a greedy approach. Without this prefix free code the decoding will fail.

5a. Best case for RLC is when N = 2^(B)-1 for B/((2^B)-1. This is because it can generate up to 256 characters. And it is the best when we just have a bunch of 1's or 0s

5b The worst case for huffman coding probability of a symbol exceeds 2^-1 = 0.5. It also suffers from the fibonacci related frequencies since it will be an unbalanced tree. The running time for compression will be N + BlogB

6. 41 42 81 43 44 46 82 83 44 80