127. Word Ladder

int ladderLength(string beginWord, string endWord, vector<string>& wordList) {

// We are creating a set out of word list totake care of any duplicates.

unordered\_set<string> dict(wordList.begin(), wordList.end());

// If the target string is not present in the dictionary, there is no such transformation sequence

if (dict.find(endWord) == dict.end()) {

return 0;

}

queue<string> Q;

Q.push(beginWord);

int steps = 0;

while(!Q.empty()) {

steps++;

int sz = Q.size();

for(int i = 0; i < sz; i++){

string cur = Q.front();

Q.pop();

for(int i = 0; i < cur.size(); i++) {

char currentChar = cur[i];

for(char c = 'a'; c <= 'z'; c++) {

cur[i] = c;

if(cur == endWord)

return steps + 1;

if(dict.find(cur) != dict.end()) {

Q.push(cur);

dict.erase(cur); // word is processed, remove it from dictionary

}

}

cur[i] = currentChar;

}

}

}

return 0;

}