TASK 1:

1) Count the occurrences of each letter in the text.

Algorithm:

1. Create an empty dictionary.
2. Update values when it finds the same character

Code:

freq = {}  
# A  
for i in teststr:  
 if i in freq:  
 freq[i] += 1  
 else:  
 freq[i] = 1

Output:

Count of all characters:

{'L': 11, 'e': 619, 't': 410, 'm': 125, 'n': 286, 'o': 345, 'h': 347, 'a': 295, 'r': 259, 'i': 302, 'g': 83, 'f': 101, 'u': 133, 'd': 194, 's': 337, 'A': 21, 'p': 68, 'l': 212, 'v': 57, 'W': 21, 'c': 72, 'w': 111, '(': 1, ')': 1, 'O': 5, 'b': 78, 'x': 4, 'è': 1, 'k': 34, 'T': 26, 'I': 19, 'y': 116, "'": 5, 'B': 11, ',': 96, '’': 34, 'z': 3, '-': 9, 'H': 2, '‘': 2, 'S': 11, 'P': 2, '.': 15, 'N': 5, 'F': 4, 'D': 5, 'C': 5, 'M': 6, '?': 2, 'j': 3, ':': 7, 'R': 2, ';': 2, 'Y': 1, 'q': 2, 'G': 1}

2) Print the number of one-letter, two-letter, three-letter words and so on.

Algorithm:

1. Let counts of each letter word.
2. Initialize them with zero
3. Check lengths of words and update according to it

Code:

count1 = 0;  
count2 = 0;  
count3 = 0;  
count4 = 0;  
count5 = 0;  
count6 = 0;  
count7 = 0;  
count8 = 0;  
count9 = 0;  
count10 = 0;  
for j in data:  
 if len(j) == 1:  
 count1 = count1 + 1;  
 elif len(j) == 2:  
 count2 = count2 + 1;  
 elif len(j) == 3:  
 count3 = count3 + 1;  
 elif len(j) == 4:  
 count4 = count4 + 1;  
 elif len(j) == 5:  
 count5 = count5 + 1;  
 elif len(j) == 6:  
 count6 = count6 + 1;  
 elif len(j) == 7:  
 count7 = count7 + 1;  
 elif len(j) == 8:  
 count8 = count8 + 1;  
 elif len(j) == 9:  
 count9 = count9 + 1;  
 elif len(j) == 10:  
 count10 = count10 + 1;  
  
print("One letter word " + str(count1))  
print("Two letter word " + str(count2))  
print("Three letter word " + str(count3))  
print("Four letter word " + str(count4))  
print("Five letter word " + str(count5))  
print("Six letter word " + str(count6))  
print("Seven letter word " + str(count7))  
print("Eight letter word " + str(count8))  
print("Nine letter word " + str(count9))  
print("Ten letter word " + str(count10))

Output :

One letter word 21

Two letter word 151

Three letter word 171

Four letter word 216

Five letter word 131

Six letter word 76

Seven letter word 65

Eight letter word 55

Nine letter word 26

Ten letter word 29

3) Print the number of occurrences of each different word in the text.

Algorithm:

1. Create an empty tuple
2. Count words and if word is found increase the count

Code:

counts = dict()  
for word in data:  
 if word in counts:  
 counts[word] += 1  
 else:  
 counts[word] = 1  
  
print(counts)

Output:

{'Let': 1, 'me': 4, 'not': 14, 'to': 22, 'the': 28, 'marriage': 1, 'of': 23, 'true': 4, 'minds': 1, 'Admit': 1, 'impediments': 1, 'love': 4, 'is': 12, 'Which': 1, 'alters': 2, 'when': 5, 'it': 6, 'alteration': 1, 'finds': 1, '(A)': 1, 'Or': 1, 'bends': 1, 'with': 5, 'remover': 1, 'remove': 1, 'no': 4, 'an': 4, 'ever': 2, 'fixèd': 1, 'mark': 1, 'That': 1, 'looks': 1, 'on': 5, 'tempests': 1, 'and': 17, 'never': 2, 'shaken': 1, 'It': 1, 'star': 1, 'every': 3, "wand'ring": 1, 'bark': 1, 'Whose': 1, "worth's": 1, 'unknown': 1, 'although': 1, 'his': 13, 'height': 1, 'be': 8, "takenLove's": 1, "time's": 1, 'fool': 1, 'though': 1, 'rosy': 1, 'lips': 1, 'cheeks': 1, 'Within': 1, 'bending': 1, "sickle's": 1, 'compass': 1, 'come': 1, 'Love': 1, 'brief': 1, 'hours': 1, 'weeks': 1, 'But': 1, 'bears': 1, 'out': 1, 'even': 1, 'edge': 1, 'doom': 1, 'If': 1, 'this': 8, 'error': 1, 'upon': 1, 'proved': 1, 'I': 9, 'writ,': 1, 'nor': 1, 'man': 1, 'loveWhen': 1, 'forty': 1, 'winters': 1, 'shall': 3, 'besiege': 1, 'thy': 16, 'brow': 1, 'And': 1, 'dig': 1, 'deep': 2, 'trenches': 1, 'in': 19, 'beauty’s': 4, 'field': 1, 'Thy': 1, 'youth’s': 1, 'proud': 1, 'livery': 1, 'so': 7, 'gazed': 1, 'now': 2, 'Will': 1, 'a': 12, 'tattered': 1, 'weed': 1, 'small': 1, 'worth': 1, 'held': 1, 'Then': 1, 'being': 2, 'asked': 1, 'where': 3, 'all': 5, 'beauty': 2, 'lies': 1, 'Where': 1, 'treasure': 1, 'lusty': 1, 'days': 2, 'To': 1, 'say,': 1, 'within': 1, 'thine': 9, 'own': 1, 'sunken': 1, 'eyesWere': 1, 'all-eating': 1, 'shame,': 1, 'thriftless': 1, 'praise': 2, 'How': 1, 'much': 1, 'more': 3, 'deserv’d': 1, 'use,If': 1, 'thou': 17, 'couldst': 1, 'answer': 1, '‘This': 1, 'fair': 5, 'child': 1, 'mineShall': 1, 'sum': 1, 'my': 12, 'count,': 1, 'make': 1, 'old': 1, 'excuse,’Proving': 1, 'by': 6, 'succession': 1, 'thine.This': 1, 'were': 1, 'new': 2, 'made': 1, 'art': 3, 'old,And': 1, 'see': 2, 'blood': 1, 'warm': 1, 'feel’st': 1, 'cold.Look': 1, 'glass': 2, 'tell': 1, 'face': 2, 'viewest,Now': 1, 'time': 2, 'that': 9, 'should': 2, 'form': 3, 'another,Whose': 1, 'fresh': 1, 'repair': 1, 'if': 3, 'renewest,Thou': 1, 'dost': 1, 'beguile': 1, 'world,': 1, 'unbless': 1, 'some': 1, 'mother.For': 1, 'she': 3, 'whose': 1, 'uneared': 1, 'wombDisdains': 1, 'tillage': 1, 'husbandryOr': 1, 'who': 2, 'he': 1, 'fond': 1, 'will': 4, 'tombOf': 1, 'self-love': 1, 'stop': 1, 'posterityThou': 1, 'mother’s': 1, 'theeCalls': 1, 'back': 2, 'lovely': 2, 'April': 1, 'her': 3, 'prime,So': 1, 'through': 2, 'windows': 2, 'age': 1, 'shaltsee,Despite': 1, 'wrinkles': 1, 'golden': 1, 'time.But': 1, 'live': 1, 'remembered': 1, 'be,Die': 1, 'single': 3, 'image': 2, 'dieswiththee.Music': 1, 'hear,': 1, 'why': 1, 'hear’st': 1, 'music': 1, 'sadly?Sweets': 1, 'sweets': 1, 'war': 1, 'not,': 1, 'joy': 1, 'delights': 1, 'joy:Why': 1, 'lov’st': 1, 'ththat': 1, 'which': 2, 'receiv’st': 2, 'gladly,Or': 1, 'else': 1, 'pleasure': 1, 'annoy?If': 1, 'concord': 1, 'well-tuned': 1, 'sounds,Byunions': 1, 'married': 1, 'do': 4, 'offend': 1, 'ear,They': 1, 'but': 7, 'sweetly': 1, 'chide': 1, 'thee,': 2, 'confoundsIn': 1, 'singleness': 1, 'parts': 1, 'thoushouldst': 1, 'bearMark': 1, 'how': 1, 'one': 4, 'string': 1, 'sweet': 1, 'husband': 1, 'another,Strikes': 1, 'each': 2, 'mutual': 1, 'orderingResembling': 1, 'sire,child,and': 1, 'happy': 1, 'mother,Who': 1, 'one,': 1, 'pleasing': 1, 'note': 1, 'singWhose': 1, 'speechless': 1, 'song': 1, 'many,': 1, 'seeming': 1, 'Sings': 1, '‘Thou': 1, 'wilt': 1, 'prove': 1, 'noneIs': 1, 'for': 9, 'fear': 1, 'wet': 1, 'widow’s': 1, 'eye,That': 1, 'consum’st': 1, 'self': 1, 'lifeAh,': 1, 'issueless': 1, 'shalt': 1, 'hap': 1, 'die,The': 1, 'world': 5, 'wail': 1, 'thee': 5, 'like': 1, 'makeless': 1, 'wife,The': 1, 'widow': 2, 'andstill': 1, 'weep,That': 1, 'hast': 1, 'left': 1, 'behind,When': 1, 'private': 1, 'well': 2, 'may': 1, 'keep,By': 1, 'children’s': 1, 'eyes,': 1, 'hehusband’s': 1, 'shape': 1, 'mind:Look': 1, 'what': 3, 'unthrift': 1, 'doth': 3, 'spendShifts': 1, 'place,': 1, 'still': 2, 'enjoys': 1, 'it;But': 1, 'waste': 1, 'hath': 5, 'end,And': 1, 'kept': 1, 'unused': 1, 'user': 1, 'destroys': 1, 'it:No': 1, 'toward': 1, 'others': 1, 'bosom': 1, 'sitsThat': 1, 'himself': 1, 'such': 1, 'murd’rous': 1, 'shame': 1, 'commits.Shall': 1, 'compare': 1, 'summer’s': 2, 'dayThou': 1, 'temperateRough': 1, 'winds': 1, 'shake': 1, 'darling': 1, 'buds': 1, 'May,And': 1, 'lease': 1, 'too': 2, 'short': 1, 'date:Sometime': 1, 'hot': 2, 'eyeof': 1, 'heaven': 1, 'shines,And': 1, 'often': 1, 'gold': 1, 'complexion': 1, 'dimmed,And': 1, 'from': 4, 'sometime': 1, 'declines,By': 1, 'chance,': 1, 'ornature’s': 1, 'changing': 1, 'course': 1, 'untrimmed:But': 1, 'eternal': 2, 'summer': 1, 'fade,Nor': 1, 'lose': 1, 'possession': 1, 'ow’st,Norshall': 1, 'death': 2, 'brag': 1, 'wand’rest': 1, 'shade,When': 1, 'lines': 1, 'grow’st,So': 1, 'long': 3, 'as': 3, 'men': 3, 'can': 3, 'breathe': 1, 'or': 1, 'eyes': 5, 'see,So': 1, 'lives': 1, 'this,': 1, 'gives': 1, 'life': 2, 'thee.glass': 1, 'persuade': 1, 'am': 1, 'old,So': 1, 'youth': 1, 'are': 1, 'date,But': 1, 'time’s': 1, 'furrows': 1, 'behold,Then': 1, 'look': 1, 'expiate.For': 1, 'beautythat': 1, 'cover': 1, 'thee,Is': 1, 'seemly': 1, 'raiment': 1, 'heart,Which': 1, 'breast': 1, 'live,': 1, 'me,How': 1, 'Ithen': 1, 'elder': 1, 'than': 1, 'artO': 1, 'therefore': 1, 'thyself': 1, 'wary,As': 1, 'self,': 1, 'will,Bearing': 1, 'thyheartwhich': 1, 'keep': 1, 'charyAs': 1, 'tender': 1, 'nurse': 1, 'babe': 1, 'faring': 1, 'ill.Presume': 1, 'heart': 1, 'mine': 1, 'slain,Thogav’st': 1, 'give': 1, 'again.Mine': 1, 'eye': 2, 'played': 1, 'painter': 2, 'stelled,Thy': 1, 'itable': 1, 'heart,My': 1, 'body': 1, 'frame': 1, 'wherein': 1, '': 1, 'held,And': 1, 'perspective': 1, 'best': 1, 'painter’s': 1, 'art.For': 1, 'mustyou': 1, 'seehis': 1, 'skill,To': 1, 'find': 1, 'your': 1, 'pictured': 1, 'lies,Which': 1, 'bosom’s': 1, 'shop': 1, 'hanging': 1, 'still,That': 1, 'windowglazed': 1, 'eyesNow': 1, 'good': 1, 'turns': 1, 'have': 2, 'done,Mine': 1, 'drawn': 1, 'shape,': 1, 'meAre': 1, 'breast,': 1, 'where-through': 1, 'sunDelights': 1, 'peep,': 1, 'gaze': 1, 'therein': 1, 'thee;Yet': 1, 'cunning': 1, 'want': 1, 'grace': 1, 'their': 1, 'art,They': 1, 'draw': 1, 'they': 1, 'see,': 1, 'know': 1, 'heart.Cupid': 1, 'laid': 1, 'brand': 3, 'fell': 1, 'asleep,A': 1, 'maid': 1, 'Dian’s': 1, 'advantage': 1, 'found,And': 1, 'love-kindling': 1, 'fire': 5, 'did': 1, 'quickly': 1, 'steepIn': 1, 'cold': 1, 'valley-fountain': 1, 'ground:Which': 1, 'borrowed': 1, 'holy': 1, 'Love,A': 1, 'dateless': 1, 'lively': 1, 'heat': 2, 'endure,And': 1, 'grew': 1, 'seeting': 1, 'bath': 4, 'yet': 1, 'prove,Againststrange': 1, 'maladies': 1, 'sovereign': 1, 'cure:But': 1, 'at': 1, 'mistress’': 2, 'Love’s': 2, 'new-fired,The': 1, 'boy': 1, 'trial': 1, 'needs': 1, 'would': 1, 'touch': 1, 'breast,I': 1, 'sick': 1, 'withal': 1, 'help': 2, 'desired,And': 1, 'thither': 1, 'hied': 1, 'sad': 1, 'distempered': 1, 'guest.But': 1, 'found': 1, 'cure,': 1, 'lies,Where': 1, 'Cupid': 1, 'got': 1, 'mThe': 1, 'little': 1, 'Love-god': 1, 'lying': 1, 'once': 1, 'asleep,Laid': 1, 'side': 1, 'heart-inflaming': 1, 'brand,Whilst': 1, 'many': 2, 'nymphs': 1, 'vowed': 1, 'chaste': 1, 'keep,Came': 1, 'tripping': 1, 'by,': 1, 'maiden': 1, 'hand,The': 1, 'fairest': 1, 'votary': 1, 'took': 2, 'up': 1, 'fire,Which': 1, 'legions': 1, 'hearts': 1, 'had': 1, 'warmed,And': 1, 'general': 1, 'desire,Was': 1, 'sleeping': 1, 'virgin': 1, 'hand': 1, 'disarmed.This': 1, 'quenched': 1, 'cool': 1, 'by,Which': 1, 'perpetual,Growing': 1, 'healthfulremedy,For': 1, 'discased,': 1, 'thrall,Came': 1, 'there': 1, 'cure': 1, 'prove,Love’s': 1, 'heats': 1, 'water,': 1, 'water': 1, 'cools': 1, 'love.': 1}

Task 2:

Find the fewest words that will link them.

Algorithm:

1. Convert list into set for duplicate values
2. If end word is not in set return 0;
3. Create a dictionary ‘curr’. Initialize with start word.
4. Initialize res (answer)
5. Create an empty tuple next
6. For each word check if there exists a word which have just one letter difference
7. Update the res
8. Add word to next

Code:

def minladder(beg, end, list):  
 *"""* ***:type*** *beginWord: object  
 """* list = set(list)  
 if end not in list:  
 return 0  
  
 curr = {beg}  
 res = 1  
 while curr:  
 list -= curr  
 next = set()  
 for word in curr:  
 for i in range(len(word)):  
 for c in 'abcdefghijklmnopqrstuvwxyz':  
 new = word[:i] + c + word[i + 1:]  
 if new == end:  
 return 1 + res  
 if new in list:  
 next.add(new)  
 curr = next  
 res += 1  
 return 0  
  
  
Lists = open('Words.txt').read().splitlines()  
  
start1 = "flour"  
target1 = "bread"  
start2 = "chaos"  
target2 = "peace"  
start3 = "river"  
target3 = "shore"  
start4 = "sleep"  
target4 = "dream"  
start5 = "black"  
target5 = "white"  
start6 = "witch"  
target6 = "fairy"  
start7 = "tears"  
target7 = "smile"  
start8 = "which"  
target8 = "think"  
start9 = "paper"  
target9= "story"  
start10 = "early"  
target10 = "trees"  
  
  
print(minladder(start1, target1, Lists))  
print(minladder(start2, target2, Lists))  
print(minladder(start3, target3, Lists))  
print(minladder(start4, target4, Lists))  
print(minladder(start5, target5, Lists))  
print(minladder(start6, target6, Lists))  
print(minladder(start7, target7, Lists))  
print(minladder(start8, target8, Lists))  
print(minladder(start9, target9, Lists))  
print(minladder(start10, target10, Lists))

Output:

7

8

9

8

8

13

7

9

10

9