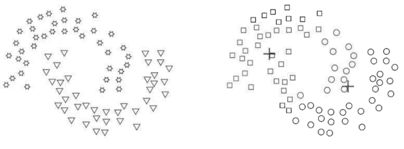
Problem 1:

1). First iteration (cluster 1: center is instance 2, cluster 2: center is instance 4)

Instance 1 is assigned to cluster 2 Instance 2 is assigned to cluster 1
Instance 3 is assigned to cluster 2 Instance 4 is assigned to cluster 2
Instance 5 is assigned to cluster 2 Instance 6 is assigned to cluster 2
Recompute the centroid for cluster 1: X: 12.0 Y:33.0 Cluster 2: X: 21.6 Y: 24.0
Second Iteration
Instance 1 is assigned to cluster 2 Instance 2 is assigned to cluster 1
Instance 3 is assigned to cluster 2 Instance 4 is assigned to cluster 2
Instance 5 Is assigned to cluster 2 Instance 6 is assigned to cluster 2
Recompute the centroid for cluster 1: X: 12.0 Y:33.0 Cluster 2: X: 21.6 Y: 24.0

No, we do not need more iterations in order to get the final clusters since the second iteration's result is the same as the first iteration's result.

2). K means clusters' usual shape is globular shape. K means did not do a good work when the clusters are of different sizes or densities or non-globular shapes. If there are outliers, K means also cannot have a proper result. For example, if the clusters are non-globular shapes, the result of K means will be not correct.

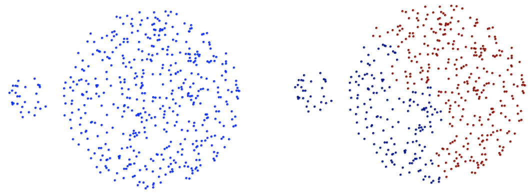


(a) Original points. (b) Two K-means clusters.

3). We can convert these points in polar coordinates. This can remove sphericity. And the non-spherical cluster can be represented as the regular shape without crossing. And this can be easily analyzed by the K-means method.

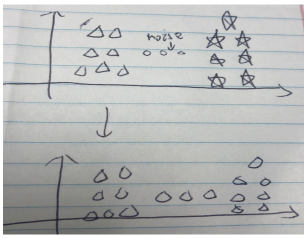
Problem 2:

It will break a large cluster when there are two cluster: one cluster is small and the other one is larger. And some points of the large cluster will be assigned to the small cluster since they are closer to that cluster. And the result will not be correct.



Original Points Two Clusters

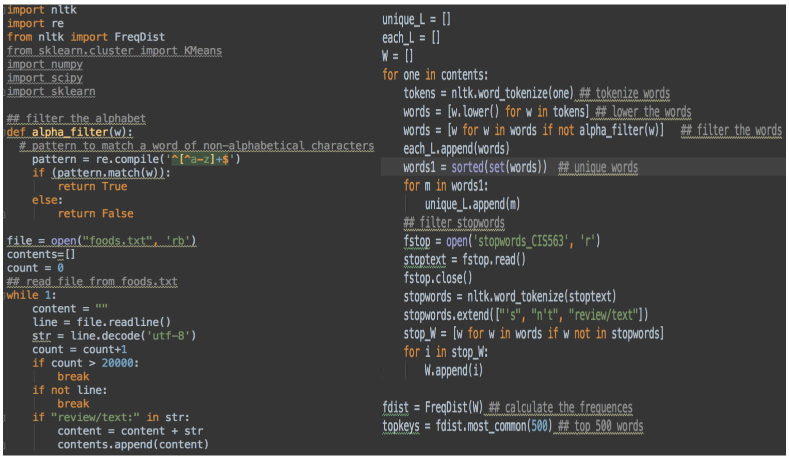
And if there are some noise which at the middle of the two cluster. The two clusters will be assigned as one cluster. Then the result will be not correct.



Problem 3:

```
[('br', 647087), ('good', 196971), ('taste', 168126), ('great', 165060), ('coffee', 162628), ('product', 148169), ('flavor', 143785),
('tea', 135147), ('love', 127015), ('will', 126793), ('food', 124926), ('amazon', 83082), ('time', 81593), ('buy', 76444), ('best',
75827), ('price', 74260), ("'m", 73832), ('find', 73226), ('well', 69584), ('better', 69368), ('dog', 69060), ('eat', 66937), ('water',
59621), ('chocolate', 59283), ('bag', 56544), ('sugar', 53103), ('cup', 50915), ('drink', 50294), ('sweet', 50222), ('bought', 49776),
('box', 49220), ('day', 46011), ('tastes', 45342), ('store', 44115), ('order', 43422), ('bit', 42607), ('recommend', 42563), ("'re",
41575), ('nice', 41130), ('delicious', 40493), ('favorite', 39698), ('flavors', 38452), ('mix', 38379), ('free', 37898), ('hot', 37506),
('cat', 37286), ('dogs', 37274), ('brand', 36955), ('stuff', 36430), ('loves', 36395), ('years', 36040), ('treats', 35672), ('lot',
35509), ('add', 35161), ('healthy', 34787), ('chips', 34414), ('ingredients', 34311), ('organic', 33871), ('quality', 33694), ('milk',
33525), ('small', 33452), ('perfect', 32446), ('ordered', 32296), ('snack', 31954), ('strong', 31829), ('eating', 31644), ('bad', 31372),
('easy', 30582), ('products', 30318), ('green', 29926), ('treat', 29842), ('hard', 29649), ('enjoy', 29517), ('fresh', 29432), ('salt',
29396), ('long', 29354), ('pack', 29218), ('bags', 29210), ('definitely', 29182), ('buying', 29130), ('oil', 28766), ('regular', 28699),
('thing', 28602), ('thought', 28526), ('cookies', 28205), ('chicken', 27785), ('high', 27608), ('pretty', 27520), ('natural', 27276),
('local', 27077), ("'d", 27050), ('size', 27047), ('work', 27023), ('happy', 26916), ('cats', 26699), ('people', 26193), ('big', 26109),
('going', 26030), ('package', 25839), ('tasty', 25353), ('diet', 25146), ('shipping', 24743), ('real', 24421), ('sauce', 24338),
('foods', 24211), ('butter', 24123), ('wonderful', 23964), ('highly', 23930), ('calories', 23907), ('feel', 23812), ('bars', 23255),
('purchase', 23198), ('tasted', 23198), ('purchased', 23181), ('dry', 23111), ('excellent', 22768), ('worth', 22739), ('texture', 22468),
('expensive', 22019), ('year', 21669), ('amount', 21571), ('protein', 21513), ('smell', 21461), ('fat', 21342), ('rice', 21284),
('stores', 21164), ('coconut', 21130), ('tasting', 21082), ('dark', 21047), ('grocery', 20707), ('half', 20654), ('low', 20458), ('kind',
20444), ('morning', 20407), ('things', 20285), ('brands', 20195), ('bottle', 19965), ('vanilla', 19953), ('fruit', 19938), ('loved',
19638), ('family', 19520), ('received', 19505), ('blend', 19472), ('full', 19448), ('days', 19089), ('months', 18909), ('problem',
18811), ('reviews', 18754), ('three', 18746), ('cereal', 18731), ('wo', 18639), ('company', 18637), ('case', 18451), ('health', 18266),
('candy', 18179), ('black', 18161), ('bar', 18117), ('money', 17558), ('light', 17541), ('making', 17524), ('flavored', 17458),
('drinking', 17376), ('review', 17374), ('peanut', 17316), ('large', 17254), ('gluten', 17078), ('arrived', 17004), ('kids', 16689),
('times', 16615), ('top', 16463), ('item', 16432), ('husband', 16361), ('bitter', 16348), ('stars', 16319), ('http', 16306), ('variety',
16282), ('disappointed', 16135), ('extra', 15992), ('corn', 15968), ('beans', 15875), ('href=', 15873), ('/a', 15873), ('fact', 15870),
('packaging', 15849), ('save', 15732), ('decided', 15677), ('cheese', 15617), ('fine', 15516), ('started', 15492), ('honey', 15470),
('boxes', 15453), ('absolutely', 15443), ('teas', 15266), ('prefer', 15227), ('popcorn', 15225), ('smooth', 15131), ('recommended',
15112), ('deal', 15083), ('couple', 14995), ('house', 14946), ('oz', 14897), ('energy', 14839), ('cream', 14643), ('works', 14553),
('wanted', 14542), ('cheaper', 14474), ('breakfast', 14412), ('cups', 14383), ('minutes', 14375), ('baby', 14339), ('cost', 14303),
('roast', 14302), ('ago', 14226), ('meal', 14171), ('open', 14144), ('read', 13938), ('white', 13888), ('syrup', 13697), ('juice',
13650), ('powder', 13624), ('longer', 13553), ('rich', 13549), ('cans', 13467), ('help', 13371), ('bold', 13234), ('meat', 13141),
('bread', 13139), ('ordering', 13130), ('serving', 13127), ('pieces', 13040), ('red', 13008), ('amazing', 13005), ('soft', 12868),
('keurig', 12867), ('cookie', 12808), ('month', 12741), ('fast', 12728), ('k-cups', 12581), ('weight', 12462), ('week', 12358), ('spicy',
12233), ('glad', 12210), ('fan', 12208), ('quick', 12102), ('soup', 12095), ('side', 12040), ('second', 11954), ('mouth', 11838),
('likes', 11703), ('plastic', 11487), ('son', 11460), ('starbucks', 11425), ('pasta', 11404), ('gift', 11384), ('difference', 11355),
('beef', 11306), ('ginger', 11263), ('hair', 11217), ('cinnamon', 11184), ('crunchy', 10982), ('friends', 10976), ('potato', 10901),
('mixed', 10901), ('opened', 10891), ('wheat', 10854), ('problems', 10824), ('jerky', 10773), ('feed', 10765), ('type', 10762),
('canned', 10742), ('ice', 10727), ('brew', 10718), ('life', 10663), ('cold', 10652), ('dried', 10546), ('market', 10492), ('online',
10489), ('reason', 10454), ('version', 10387), ('teeth', 10370), ('pet', 10326), ('easily', 10277), ('super', 10247), ('nuts', 10226),
('enjoyed', 10178), ('chew', 10169), ('fiber', 10153), ('weeks', 10145), ('artificial', 10127), ('care', 10123), ('formula', 10096),
('start', 10094), ('container', 10078), ('brown', 10049), ('salty', 10049), ('cooking', 10033), ('smaller', 10026), ('machine', 10011),
('cut', 9973), ('daughter', 9971), ('left', 9956), ('guess', 9927), ('yummy', 9924), ('french', 9914), ('k-cup', 9821), ('snacks', 9816),
('soy', 9802), ('vet', 9684), ('course', 9671), ('drinks', 9662), ('plain', 9651), ('place', 9648), ('blue', 9629), ('recipe', 9598),
('awesome', 9550), ('exactly', 9539), ('ingredient', 9524), ('night', 9498), ('bowl', 9486), ('choice', 9425), ('spice', 9378), ('wife',
9376), ('coffees', 9368), ('oatmeal', 9352), ('clean', 9345), ('alternative', 9326), ('single', 9324), ('lemon', 9282), ('original',
9259), ('daily', 9250), ('surprised', 9174), ('simply', 9146), ('aroma', 9125), ('cocoa', 9115), ('pleased', 9097), ('crackers', 9084),
('wrong', 9080), ('pay', 9061), ('special', 9049), ('ate', 9045), ('decaf', 9045), ('flour', 9008), ('aftertaste', 8965), ('smells',
8948), ('hand', 8948), ('sodium', 8941), ('list', 8932), ('compared', 8924), ('friend', 8871), ('pods', 8866), ('color', 8861), ('body',
8858), ('cook', 8848), ('caffeine', 8843), ('almonds', 8781), ('finally', 8775), ('close', 8772), ('picky', 8762), ('inside', 8721),
('mild', 8712), ('jar', 8704), ('subscribe', 8700), ('huge', 8698), ('hope', 8662), ('heat', 8631), ('experience', 8614), ('change',
8611), ('noticed', 8603), ('feeding', 8596), ('ground', 8553), ('hours', 8547), ('flavorful', 8523), ('looked', 8520), ('chip', 8517),
('stick', 8511), ('stomach', 8504), ('apple', 8494), ('pepper', 8386), ('service', 8374), ('idea', 8368), ('helps', 8352), ('orange',
8304), ('live', 8286), ('packs', 8264), ('needed', 8207), ('fish', 8167), ('soda', 8148), ('cake', 8137), ('delivery', 8126),
('expected', 8125), ('adding', 8102), ('gum', 8065), ('leave', 8064), ('bite', 8061), ('lots', 8054), ('larger', 8020), ('espresso',
7980), ('leaves', 7947), ('instant', 7944), ('bottles', 7910), ('completely', 7909), ('star', 7875), ('seeds', 7851), ('bulk', 7828),
('it.', 7824), ('raw', 7810), ('convenient', 7808), ('continue', 7807), ('noodles', 7792), ('fantastic', 7788), ('grain', 7769), ('skin',
7768), ('iced', 7751), ('bottom', 7731), ('medium', 7726), ('takes', 7724), ('weak', 7665), ('cheap', 7657), ('extremely', 7562),
('packaged', 7539), ('mind', 7524), ('expect', 7501), ('rest', 7500), ('eaten', 7453), ('mine', 7440), ('label', 7411), ('chewy', 7387),
('pot', 7387), ('wellness', 7381), ('packages', 7378), ('healthier', 7376), ('shipped', 7345), ('excited', 7294), ('stock', 7284),
('chai', 7280), ('ounce', 7275), ('granola', 7245), ('told', 7242), ('true', 7235), ('based', 7208), ('glass', 7192), ('filling', 7143),
('waste', 7142), ('issues', 7137), ('packets', 7134), ('sold', 7126), ('maker', 7123), ('grams', 7080), ('items', 7037), ('normal',
7033), ('entire', 7006), ('crunch', 6938), ('note', 6936), ('opinion', 6905), ('packet', 6892), ('carry', 6864), ('thick', 6853),
('point', 6833), ('calorie', 6832), ('eats', 6817), ('wait', 6806), ('called', 6756), ('content', 6754), ('pound', 6746), ('today',
6720), ('sweetness', 6719), ('hint', 6698), ('purchasing', 6693), ('almond', 6686), ('puppy', 6678), ('baking', 6606), ('mountain',
6570), ('liquid', 6563), ('lunch', 6562), ('olive', 6515), ('stash', 6509), ('reading', 6504), ('bean', 6497), ('easier', 6457),
('break', 6423), ('vitamin', 6419), ('creamy', 6413), ('difficult', 6389), ('christmas', 6381), ('pop', 6374), ('pure', 6367),
('delivered', 6364), ('tiny', 6345), ('dinner', 6319), ('sale', 6275), ('decent', 6248), ('benefits', 6242), ('stopped', 6228),
('customer', 6160), ('mint', 6158), ('worked', 6109), ('lower', 6105), ('varieties', 6087), ('loose', 6078), ('batch', 6067)]
```

```
[[('good', 196971), ('coffee', 162628), ('taste', 168126), ('well', 69584), ('tastes', 45342)], [('br', 647087), ('food', 124926), ('taste',
168126), ('product', 148169), ('good', 196971)], [('br', 647087), ('good', 196971), ('taste', 168126), ('product', 148169), ('tea', 135147)],
[('br', 647087), ('stuff', 36430), ('sauce', 24338), ('hot', 37506), ('bread', 13139)], [('great', 165060), ('br', 647087), ('flavor',
143785), ('taste', 168126), ('product', 148169)], [('kind', 20444), ('ordered', 32296), ('tasty', 25353), ('white', 13888), ('dog', 69060)],
[('treats', 35672), ('problems', 10824), ('br', 647087), ('br', 647087), ('br', 647087)], [('br', 647087), ('coffee', 162628), ('sweet',
50222), ('hot', 37506), ('cold', 10652)], [('br', 647087), ('good', 196971), ('coffee', 162628), ('taste', 168126), ('flavor', 143785)],
[('coffee',      162628),      ('br',      647087),      ('good',      196971),      ('flavor',      143785),      ('taste',      168126)]]
```



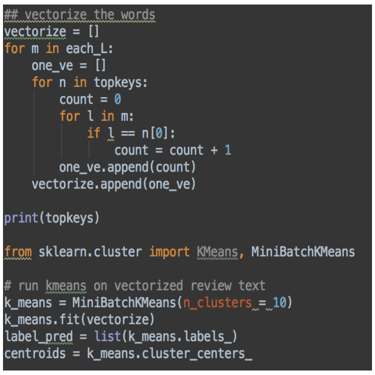
```
import nltk
import re
from nltk import FreqDist
from sklearn.cluster import KMeans
import numpy
import scipy
import sklearn

## filter the alphabet
def alpha_filter(w):
  # pattern to match a word of non-alphabetical characters
    pattern = re.compile('^[^a-z]+$')
    if (pattern.match(w)):
        return True
    else:
        return False

file = open("foods.txt", 'rb')
contents=[]
count = 0
## read file from foods.txt
while 1:
    content = ""
    line = file.readline()
    str = line.decode('utf-8')
    count = count+1
    if count > 20000:
        break
    if not line:
        break
    if "review/text:" in str:
        content = content + str
        contents.append(content)
```

```
unique_L = []
each_L = []
W = []
for one in contents:
    tokens = nltk.word_tokenize(one) ## tokenize words
    words = [w.lower() for w in tokens] ## lower the words
    words = [w for w in words if not alpha_filter(w)]   ## filter the words
    each_L.append(words)
    words1 = sorted(set(words))  ## unique words
    for m in words1:
        unique_L.append(m)
    ## filter stopwords
    fstop = open('stopwords_CIS563', 'r')
    stoptext = fstop.read()
    fstop.close()
    stopwords = nltk.word_tokenize(stoptext)
    stopwords.extend(["'s", "n't", "review/text"])
    stop_W = [w for w in words if w not in stopwords]
    for i in stop_W:
        W.append(i)

fdist = FreqDist(W) ## calculate the frequences
topkeys = fdist.most_common(500) ## top 500 words
```

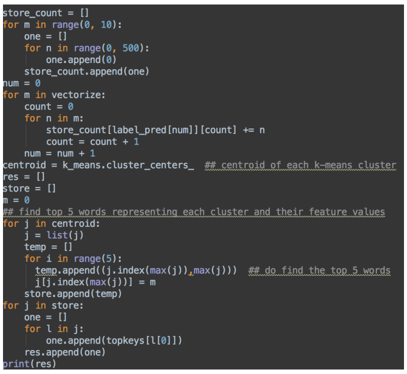


```
## vectorize the words
vectorize = []
for m in each_L:
    one_ve = []
    for n in topkeys:
        count = 0
        for l in m:
            if l == n[0]:
                count = count + 1
        one_ve.append(count)
    vectorize.append(one_ve)

print(topkeys)

from sklearn.cluster import KMeans, MiniBatchKMeans

# run kmeans on vectorized review text
k_means = MiniBatchKMeans(n_clusters = 10)
k_means.fit(vectorize)
label_pred = list(k_means.labels_)
centroids = k_means.cluster_centers_
```



```
store_count = []
for m in range(0, 10):
    one = []
    for n in range(0, 500):
        one.append(0)
    store_count.append(one)
num = 0
for m in vectorize:
    count = 0
    for n in m:
        store_count[label_pred[num]][count] += n
        count = count + 1
    num = num + 1
centroid = k_means.cluster_centers_  ## centroid of each k-means cluster
res = []
store = []
m = 0
## find top 5 words representing each cluster and their feature values
for j in centroid:
    j = list(j)
    temp = []
    for i in range(5):
      temp.append((j.index(max(j)),max(j)))  ## do find the top 5 words
      j[j.index(max(j))] = m
    store.append(temp)
for j in store:
    one = []
    for l in j:
        one.append(topkeys[l[0]])
    res.append(one)
print(res)
```

Problem 4:

4. (a). ζ is an anti-monotone measure.

Since $\zeta(\{A,B\}) = \min\,(c(A \to B, c(B \to A)) = \frac{s(A,B)}{\max\,(s(A),s(B))}$ $\zeta(\{A,B,C\}) = \min(c(A \to BC), c(B \to AC), c(C \to AB)) = \frac{s(A,B,C)}{\max\,(s(A),s(B),s(C))}$

And we can know that $s(A,B,C) \le s(A,B)$. And $\max\,(s(A),s(B),s(C)) \ge \max\,(s(A),s(C))$. Then $\zeta(\{A,B\}) \ge \zeta(\{A,B,C\})$. So ζ is an anti-monotone measure.

(b). η is non-monotone.

Since $\eta(\{A,B\}) = \min(\mathrm{c(A \to B), c(B \to A)}) = \frac{s(A,B)}{\max\,(s(A),s(B))}$ $\eta(\{A,B,C\}) = \min(c(AB \to C), c(AC \to B), c(BC \to A)) = \frac{s(A,B,C)}{\max\,(s(A,B),s(A,C),s(B,C))}$

And we know that $s(A,B,C) \le s(A,B)$, $\max(s(A,B),s(A,C),s(B,C)) \le \max\,(s(A),s(B))$. Then we cannot judge which one is bigger or equal. So η is non-monotone.

(c). We can assume that $\zeta 1(\{A_1, A_2, \ldots A_k\}) = \max\,(c(A_1 -> A_2, A_3, \ldots A_k), \ldots c(A_k \to A_1, A_2, \ldots, A_{k-1}))$. Then $\zeta 1(\{A,B\}) = \max\,(c(A \to B, c(B \to A)) = \frac{s(A,B)}{\min\,(s(A),s(B))}$. And $\zeta 1(\{A,B,C\}) = \max(c(A \to BC), c(B \to AC), c(c \to AB)) = \frac{s(A,B,C)}{\min\,(s(A),s(B),s(C))}$.

Then $s(A,B,C) \le s(A,B)$, $\min(s(A),s(B),s(C)) \le \min(s(A),s(B))$. Then we cannot judge which one is bigger or they are equal. So $\zeta 1$ is non-monotone.

We can assume $\eta 1(\{A_1, A_2, \ldots A_k\}) = \max\,(c(A_2, A_3, \ldots A_k \to A_1), \ldots c(A_1, A_2, \ldots, A_{k-1} \to A_k))$. Then $\eta 1(\{A,B\}) = \max(c(A \to B), c(B \to A)) = \frac{s(A,B)}{\min\,(s(A),s(B))}$. And $\eta 1(\{A,B,C\}) = \max(c(AB \to C), c(AC \to B), c(BC \to A)) = \frac{s(A,B,C)}{\min\,(s(A,B),s(A,C),s(B,C))}$.

Then we can get to know $s(A,B,C) \le s(A,B)$ and $\min\,(s(A,B),s(A,C),s(B,C)) \le \min\,(s(A),s(B))$. Then we cannot judge which one is bigger or they are equal. So $\eta 1$ is non-monotone.

6. (a). We can get to know that there are six items in the data set. Then the total number of rules is 602 according to the association rule calculation formula.

(b). Since we can get that the longest transaction contains about 4 items. And then the maximum size of the frequent itemset is 4.

(c). We can get to know that the maximum number of size-3 itemset can be calculated by getting 3 from the 6 items. That is $\binom{6}{3} = 20$.

(d). The itemset which has the largest support is {Bread, Butter}.

(e). {Bread, Butter} have the same confidence which {a}->{b} and {b}->{a}.

9. (a). When finding the candidates of the transaction, the visited left nodes are: L1, L3, L5, L9 and L11.

(b). The candidates contained in the transaction are {1,4,5} (in L1), {1,5,8}(in L3) and {4,5,8}(L3).

Problem 5

For an item to be frequent, it should have minimum support 30% and minimum confidence 30%.

For itemset of size 1:

{E} = 3 {Q} = 3 {R} = 4 {T} = 4 {W} = 5 {Y} = 3

Then are all frequent.

For itemset of size 2:

{EQ} = 1 {ER} = 3 {ET} = 1 {EW} = 2 {EY} = 2

{QR} = 2 {QT} = 2 {QW} = 2 {QY} = 0

{RT} = 2 {RW} = 3 {RY} = 2

{TW} = 4 {TY} = 2

{WY} = 3

Then {ER}, {EW}, {EY}, {QR}, {QT}, {QW}, {RT}, {RW}, {RY}, {TW}, {TY}, {WY} are frequent.

For itemset of size 3:

{ERT} = 1 {ERW} = 2 {ERY} = 2

{EWY} = 2

{QRT} = 1 {QRW} = 1 {QRY} = 0

{QTW} = 2

{RTW} = 2 {RTY} = 1

{RWY} = 2

{TWY} = 2

Then {ERW}, {ERY}, {EWY}, {QTW}, {RTW}, {RWY}, {TWY} are frequent.

{ERWY} = 2

{RTWY} = 1

Then {ERWY} is frequent.

And there are no itemset of size 5.

2).

Itemset	support	confidence
{ER}->W	2	2/3
{EW}->R	2	2/2
{RW}->E	2	2/3
{ER}->Y	2	2/3
{EY}->R	2	2/2
{YR}->E	2	2/2

{EW}->Y	2	2/2
{EY}->W	2	2/2
{YW}->E	2	2/3
{QT}->W	2	2/2
{QW}->T	2	2/2
{WT}->Q	2	2/4
{RT}->W	2	2/2
{RW}->T	2	2/3
{WT}->R	2	2/4
{RW}->Y	2	2/3
{RY}->W	2	2/2
{WY}->R	2	2/3
{TW}->Y	2	2/4
{TY}->W	2	2/2
{YW}->T	2	2/3