Xiaonan Hu & Liqi Zhu's Project 1

Task1

```
cat census-dist-female-first.txt census-dist-male-first.txt >census-dist-all-first.txt
tr -sc 'A-Za-z' '\n' < census-dist-all-first.txt > all.first.txt
tr '[A-Za-z]' 'c' < all.first.txt | sort | uniq -c | sort -r > c-frequency.txt
awk '{printf ("%s \t",$1);printf("%s \n",expr length($2))}' c-frequency.txt
```

No.	Length	Frequency
1	6	1438
2	5	1221
3	7	1120
4	4	654
5	8	536
6	9	217
7	3	216
8	10	52
9	2	33
10	11	7

Results shown in the table above.

According to the display, length of **6** is the most frequent and length of **11** is the least frequent.
Simply measuring the length of letters uses little information of the dataset. For example, names origins from other languages tend to have difference length distribution, this kind of region difference make the former results mean less.

Task2

```
tr -sc '^.*[AEIOUY]+.*$' '\n' < all.first.txt | sort | uniq
# all combinations of vowels
tr -s '^.*[AEIOUY]+.*$' '\n' < all.first.txt | sort | uniq
# all combinations of consonants
cp all.first.txt all.firstcopy.txt
sed -i 's/GUE\b/c/' all.firstcopy.txt
sed -i 's/QUE\b/c/' all.firstcopy.txt
sed -i 's/ION\b/vc/' all.firstcopy.txt
sed -i 's/DGE\b/c/' all.firstcopy.txt
sed -i 's/THE\b/c/' all.firstcopy.txt
sed -i 's/LE\b/c/' all.firstcopy.txt
sed -i 's/AL\b/c/' all.firstcopy.txt
sed -i 's/EL\b/c/' all.firstcopy.txt
sed -i 's/UL\b/c/' all.firstcopy.txt
# replace endings of GUE\QUE\ION\GE\DGE\THE\LE\AL\EL\UL
egrep '[^AEIOU]E$' < all.firstcopy.txt | sed -i 's/E\b//' all.firstcopy
.txt
# delete remaining E$s behind consonants
egrep '[AEIOU]R[AEIOU]' < all.firstcopy.txt
sed 's/[AEIOU]R[AEIOU]/vcv/g' < all.firstcopy.txt > v.txt
# if [AEIOU]R[AEIOU] -> replace with 'vcv'
sed 's/\
(B\|BB\|BT\|C\|CK\|CH\|CC\|D\|DD\|LD\|F\|FF\|G\|GG\|GU\|GH\|GN\|HH\|J\|JJ\
|K\|KK\|KN\|L\|LL\|LF\|M\|MM\|N\|NN\|NG\|P\|PH\|PS\|PP\|PN\|Q\|QQ\|R\|RR\|
S\|SH\|SS\|SC\|T\|BT\|GHT\|TT\|TH\|TCH\|V\|VV\|W\|WW\|WH\|X\|XX\|Z\|ZZ\)/c
/g' v.txt > c.txt
# replace repeat consonants and combination
sed 's/\bH/c/' c.txt | sed 's/H//g' | sed 's/[BCDFGJKLMNPQRSTVWXZ]/c/g'
> v2.txt
# determine H and the remaining consonants
sed 's/\
(A\|AE\|AY\|AI\|AW\|AU\|AUGH\|E\|EE\|EW\|EA\|EAU\|EI\|EY\|EU\|EIGH\|EO\|I\
|IE\|II\|IGH\|O\|OE\|OA\|OW\|OU\|OO\|OUGH\|OI\|OY\|U\|UY\|UR\|UE\|UI\|Y\)/
v/g' v2.txt > Syllable.txt | sort | uniq -c | sort -r
# replace vowels > Syllable.txt
egrep '[A-Z]' Syllable.txt
#check
cat Syllable.txt | sort | uniq -c | sort -r | head -n20
```

No.	Structures	Frequency
1	cvcv	874
2	cvcvc	541
3	cvcvcv	476
4	cvc	342
5	cvccv	305

Top 5 frequent syllable structures shown in the table above.

'**CVCV**' is the most frequent and there are many structures with appearance of 1 such as ccvccvvcv, cccvcv, etc..
Our algrithom firstly replace the special words boundaries pattern, then delete silent endings, replace the normal patterns of consonants and vowels with c and v. It's fast and easy to adapt in other datasets.
Shortcoming of this method is that we can only identify the sequences themselves rather than in the context. We had a brief view on the raw data, there are some names obviously origin from other languages, they may pronunce differently with the same character. English names themselves don't follow the pronunciation rules sometimes eigther.
Our algorithm's shortcoming is that we weren't able to build a definition of each character's pronunciation due to absence of linguistics knowledge. We manually define those patterns of consonants and vowels, so our results do have some errors of those words with specical pronunciations or sequences.

Task 3

```
sed 's/\(DE\b\|QUE\b\|GUE\b\|GG\b\)/stops_/' all.first.txt > phonic1.tx
t
#ending stops
sed -i 's/\(CE\b\|SE\b\|\THE\b\|VE\b\|FE\b\|ZE\b|THE\b\)/fricatives_/'
phonic1.txt
#ending fricatives
sed -i 's/\(SION\b\|CION\b\|TION\b\)/fricativesION_/' phonic1.txt
#ending Fricatives [SION CION TION]
sed -i 's/\(\bX\|\bH\)/fricatives_/' phonic1.txt
#Starting Fricativess[H,X]
sed -i 's/\bWHO/fricativesO_/' phonic1.txt
#Starting Fricativess[WHo]
#
sed -i 's/\(TURE\b\)/affricates_URE/' phonic1.txt #ending
Affricates[TURE]
```

```
sed -i 's/\(GE\b\|DGE\b\)/affricates_/' phonic1.txt #ending Affricates
sed -i 's/\bGE/affricates_E/' phonic1.txt #Starting Affricates[GE]
sed -i 's/\(ME\b\|GN\b\|NE\b\)/nasals_/' phonic1.txt #ending
nasals[ME|GN|NE]
sed -i 's/\(LE\b\|AL\b\|EL\b\|UL\b\|IL\b\|LL\b\)/nasals_/' phonic1.txt
#ending liquids
sed -i 's/\bU/glides_/' phonic1.txt #starting Glides
sed 's/\
(B\|BB\|BT\|GHT\|C\|CK\|CC\|D\|DD\|LD\|G\|GU\|GH\|K\|KK\|P\|PP\|Q\|QQ\|T\|
TT\|GHT\)/stops_/g' phonic1.txt > phonic2.txt
#normal stops pattern
sed -i 's/\
(F\|FF\|LF\||P\|PH\|PS\|S\|SS\|SC\|SH\|TH\|TCH\|V\|VV\|Z\|ZZ\)/fricatives_
/g' phonic2.txt
#normal Fricatives pattern
sed -i 's/\(CH\|TCH\|GG\|J\|JJ\)/affricates_/g' phonic2.txt
#normal Affricates pattern
sed -i 's/\(M\|MM\|MB\|MN\|N\|NN\|KN\|PN\|GN\|EN\|AN\|NG\)/nasals_/g' p
honic2.txt
#normal Nasals pattern
sed -i 's/\(L\|LL\|R\|RR\|WR\|RH\)/liquids_/g' phonic2.txt
#normal Liquids pattern
sed -i 's/\(W\|WH\|UI\|IO\)/glides_/g' phonic2.txt
#normal Glides pattern
sed -i 's/H//g' phonic2.txt
#delete all the remianing silent Hs
sed -i 's/X/stopsfricative_/g' phonic2.txt
#replace all the remaining Xs.
egrep '[BCDFGHJKLMNPQRSTVWXZ]' phonic2.txt
#check to see if there's remaining Manner of articulations.

sed 's/\(EE\|OO\|\bY\)/high_/' phonic2.txt > phonic.txt
#sepcial vowel pattern
sed -i 's/A[AEIOU]\?/low_/g' phonic.txt
sed -i 's/E[AEIOU]\?/mid_/g' phonic.txt
sed -i 's/I[AEIOU]\?/high_/g' phonic.txt
sed -i 's/O[AEIOU]\?/mid_/g' phonic.txt
sed -i 's/U[AEIOU]\?/high_/g' phonic.txt
#replace vowels
sed -i 's/low_Y$/low_/' phonic.txt
sed -i 's/mid_Y$/mid_/' phonic.txt
#delete the ending Ys whose pronanciation follows the previous low/mid
vowels
sed -i 's/Y/high_/' phonic.txt

```

```
egrep [A-Z] phonic.txt
#check
cat phonic.txt | sort | uniq -c | sort -r | head -n20
```

No.	Sequences	Frequency
1	stops_mid_liquids_high_	34
2	stops_liquids_high_fricatives_stops_nasals_	18
3	stops_mid_stops_high_	17
4	stops_low_liquids_high_	16
5	stops_mid_stops_	15
5	stops_mid_liquids_low_	15
5	stops_low_stops_liquids_high_nasals_	15

Top 5 frequent sequences shown in the table above.

'**stops_mid_liquids_high_**' is the most frequent sequence, and there's many sequences with frequency of one such as '**affricates_high_liquids_high_mid_nasals_nasals_**', '**affricates_high_stops_fricatives_mid_nasals_**',etc.. And we did some further researches, we find that names begining with '**stops**' are most frequent, 1,823 in total; and names ending with '**low**' are most frequent, 1,866 in total.
Shortcoming of this method is also mainly caused by language origin differences, some names just don't pronunced as normal english.
Our algorithm's shortcoming is basically the same as Task2, we manually define the pattern for each Manner of articulation for consonants and height of vowels. Error can be those missing of judging function of each character. With this algorithm, pronunciation of some characters, such as 'c', 'g' etc., can not be precisely defined. But as we have checked some special combination of letters with 'egrep', few words are involved so we think the error is acceptable.