Math 381 Project 2:
Improve Type Efficiency by Redesigning the Keyboard

Group 7
Li Shen
Shiyao Lu
Wen Lin Tan
Wilbert Lam

03/04/2016

Contents

1 Introduction **2**

2 Background **2**

3 Model **2**
3.1 Goals . . . 2
3.2 Assumptions . . . 2
3.3 Data Sources . . . 3
3.4 Creating our Keyboards . . . 3
3.4.1 Manual method . . . 4
3.4.2 Greedy method . . . 4
3.4.3 Random method . . . 5
3.5 How Good is Our Model? . . . 5

4 Solutions **6**
4.1 Overview . . . 6
4.1.1 Terminology . . . 6
4.2 Modern Literature . . . 6
4.2.1 QWERTY keyboard . . . 6
4.2.2 MANUAL PLACEMENT keyboard . . . 6
4.2.3 GREEDY ALGORITHM keyboard . . . 7
4.2.4 RANDOM PLACEMENT keyboard . . . 7
4.3 Charles Dickens . . . 7
4.3.1 QWERTY keyboard . . . 7
4.3.2 MANUAL PLACEMENT keyboard . . . 7
4.3.3 GREEDY ALGORTHM keyboard . . . 8
4.3.4 RANDOM PLACEMENT keyboard . . . 8

5 Commentary **8**

6 Variations **9**
6.1 LOL haaiii REDdit! AMA! . . . 9
6.1.1 QWERTY Keyboard . . . 9
6.1.2 ALPHA Keyboard . . . 10
6.1.3 New GREEDY ALGORITHM Keyboard . . . 10
6.2 Parle Français? Investigating French Texts . . . 10
6.3 *Fantastic Authors and Where to Find Them* - Let's Find J.K. Rowling . . . 12

7 Conclusion **14**

8 Appendix **15**
8.1 Java Code . . . 15
8.2 Reddit text . . . 22

1 Introduction

In this paper, we are going to redesign the current keyboard layout to improve the typing efficiency. Our goal is to type with our hands alternately as much as possible since it can moderately increase the typing speed provided that users are use to the new layout.

To design the keyboard, we are going to perform textual analysis using Markov chains, and use Java to generate a transition matrix. From that we will determine the letters that we type with left hand and the letters that we type with right hand.

2 Background

We came about this problem from the purpose that we want to design something useful and applicable in real life. And while we were brainstorming ideas for this project, Li read an article about the pros and cons of the current QWERTY keyboard [20]. Therefore he suggested using Markov chains to perform some text analysis to redesign the layout of current keyboard.

Originally we were going to actually design the new keyboard by trying to determine where each letter's exact individual position was. But later we figured it might be harder than we thought because in order to do so, we would have to rank the efficiency of each finger and further take into account the ergonomic feasibility. Therefore we eventually settled on just deciding what letters we would type with our left hand and with our right hand, instead of allocating each letter to a specific position on the standard keyboard layout.

There have been many studies conducted on text analysis using Markov chain. Here are a few studies that are similar to our project:

- A.A Markov's application to Eugeny Onegin: A.A. Markov studied the sequence of 20,000 letters in A.S. Puskin's poem "Eugeny Onegin" and reached a conclusion that the stationary vowel probability is $p = 0.432$, that the probability of a vowel following a vowel is $p_1 = 0.128$, and that the probability of a vowel following a constant is $p_2 = 0.663$ [21].
- Identifying gender of authors: in the paper the author used Markov chain to identify author gender. And overall the method did a pretty good job, better than based on a guess off a coin flip. And the author implied that determining gender has further implication,e.g., determining the genre of a text if a text fits into a particular literary genre; or help websites to tailor their services to individual based on their gender [14].

3 Model

3.1 Goals

We want to develop what we call our ALPHA keyboard, a keyboard to beat QWERTY in terms of efficiency and perhaps replace it in the future.

3.2 Assumptions

We assume the probability of two adjacent letter combinations can be modeled with a Markov chain. Therefore, we construct a transition matrix for the alphabets for each category of text. Our objective is to improve the efficiency of the QWERTY keyboard with our keyboard. We want to maximize the usage of both hands and as a result, quicken typing speed. Thus, both hands ideally need to be used alternately.

Let R be denoted as right hand and L be denoted as left hand. For example, the ideal keyboard we want is to use LRLRLRLR or RLRLRLR instead of LLLRRRLRLLLLLLRRRRLLRRR. However, there are cases

where it is inevitable that two letters are typed with the same hand. There are simply too many possibilities of letter combinations in the English language for us to guarantee alternating hand combinations on adjacent letters.

Despite the fact that we want to model the problem as realistically as possible, we still make a handful of simplifications to make our model more manageable:

1. We assume the text can be modeled using a Markov chain. The probability of adjacent letters in text is assumed to follow a Markov chain.
2. We consider only the twenty-six standard letters in the English alphabet and ignore numerical values, punctuation, white-space, and special characters.
3. We consider all letters regardless of their case as lower-case.
4. We do not consider the variation of typing with different fingers, that is, we do not rank the efficiency of each key based on which finger is used to type it. We only consider the efficiency for left and right hands, and assume they are equal (our intended users are, for most part, equally capable of typing with either hand).
5. Modern literature is assumed to be the basis for creating our keyboard model. By modern literature, we mean that the sources are published within the last two years. Due to the changes in English vernacular over the years, this allows us to investigate in a more contemporary and applicable setting.

3.3 Data Sources

We initially use two major categories of data: the above-stated modern literature and a collection of Charles Dickens' books. The Charles Dickens' books are specifically chosen because the QWERTY keyboard was developed shortly after many of his books were published. This gives us an insight into how effective QWERTY was given the texts of that time period against which it was developed alongside. Each of these texts is taken from Project Gutenberg and transferred into a plain text file:

Modern texts:

- *Prime Difference* by Alan Edward Nourse [15]
- *In Bad Company and other stories* by Rolf Boldrewood [1]
- *Break a Leg* by Jim Harmon [8]
- *A Dog Day* by Walter Emanuel [6]

Charles Dickens' texts:

- *A Christmas Carol* [3]
- *The Mystery of Edwin Drood* [5]
- *American Notes* [4]

3.4 Creating our Keyboards

Without any keyboards, there is no point in creating this model! So designing our keyboard is naturally one of the most vital steps. As stated above in Section 3.1, we've made the assumption that we consider only the separation of letter keys by left and right, irrelevant of its accessibility to each finger. Thus our keyboard model can be simply divided into a left (L) and right (R) side, with letters going into an arbitrarily layout on their respective side.

One important aspect in creating our best layouts is the idea of a transition matrix modeled from a Markov chain where the states are the twenty-six lower-case letters in the English alphabet. The probabilities from state one to state two represent the long-term likelihood of having a letter follow another letter in the English language, based off the text sample we provide. We can create this by running the text sample we desire in our code (found in the appendix). Our code goes through a sample text file and creates a two dimensional array (representing our matrix) where the row represents state one and column represents state two. For each pair of adjacent letters found in the file, the corresponding coordinate representing that state jump is then incremented. After the entire list of files has been scanned, we divide each element in our matrix by the overall letter count of all the files. This obtains a transition matrix to aid us in a couple of the methods below.

Now finding this layout can be done with multiple algorithms.

3.4.1 Manual method

Since coding this method turns out to be more complicated than expected, we do this by hand and call it the "Manual method". First we create our transition matrix for adjacent letters. Looking at letter pairs with the highest probability can help aid us in assigning the letter pairs to separate sides of the keyboard. For example, if "ae" has the most occurrences, "a" can be sorted to L and "e" can be sorted to R. However, this method breaks down as the alphabets pairs probability decreases. For example, assume the case where "a","t" and "y" are sorted to L and "n", "p", "l" are sorted to R. The next highest probability might be "at". However, based on previous higher probabilities like "an" and "tl", we prioritize "a" "n" and "t" "l" to be separated as opposed to "at" because "at" occurs with smaller probability. This continues down until we have no more letters left. However, more complication arises in coding because our algorithm intends to maximize the placement of the pairs with high probability occurrence to be L and R. For example, "ae", "ef", "fg" and "gl" will sort Left : "a", "f", "l" and Right : "e", "g". However, if the next probable pair is "ty", we cannot simply sort "t" to L or "y" to R. We need to look beyond the pairs to find the relation between the first sorted group. If the next few pairs are "yo" and "ol", we can fill in the blanks for "t", "y", "o", and "l" because the occurrence of "l" in "ol" connects "l" with "g". Therefore, we get "a", "f", "l", "r", "y" for left, and "e" ,"g", "o", "t" for right.

3.4.2 Greedy method

Another method we use to determine a potential layout for our keyboard is based off the idea of a greedy algorithm. We have two lists: one to represent the left side of the keyboard and the other the right side. The idea behind the algorithm is to assign an arbitrary starting letter (in our case "a") to the left list and then assign the next letter with the highest probability following the current letter (as listed in our transition matrix) into the opposite side list. That next letter would then be the current letter, and the process is repeated until we run out of letters. This gives us a relatively clean way of ensuring letters that frequently follow each other are at least not always clumped together on the same side of the keyboard.

The pseudo-code for the algorithm is below:

```
List left;
List right;
Set usedLetters;

left.add('a');
usedLetters.add('a“);

boolean isLeft = false;
while letters remain do
    nextLetter = getHighestPercLetterOfNextLetterNotUsed(transition_matrix);
    usedLetters.add(nextLetter);
    if isLeft then
        left.add(nextLetter)
    else
        right.add(nextLetter)
    end
end
return left,right;
```

The full code is attached in the appendix.

3.4.3 Random method

This method follows the adage that a monkey smashing keys on a keyboard will eventually spit out the complete works of Shakespeare. We randomly assign the letters to the left and right of this keyboard. This acts as somewhat of a control keyboard to show the efficiency of randomly placing characters on the keyboard. We use random number generators (RNG) in Java and assign them to the L and R arrays. Since Java characters can also be represented as integers, we can use the integer RNG in java. Our generator creates integers from 97-123, which can be cast to the equivalent ASCII value for lower-case English letters. For example, (char) 97 is “a”. Ideally, we want to see the randomly placed keyboards have worse efficiency than the keyboards we design (we want to look better than a bunch of monkeys!).

3.5 How Good is Our Model?

Having a keyboard doesn't mean anything if we don't have a way to see how effective it is. While we're representing things with a Markov chain transition matrix, we can reuse this same idea to see the percentage of time our keyboard goes between a letter from the left hand side or right hand side to the either the left or right hand side with the following adjacent letter. By choosing some text we can go and model another Markov chain with the states of left (L) and right (R). Therefore we can see the probability of going right hand to left hand (LR), right hand to right hand (RR), left hand to right hand (LR) and left hand to left hand (LL). Since this is a Markov chain, all rows in this matrix will add up to 1, so we know RR + RL = 1 and LR + LL = 1. Ideally the perfect keyboard (if theoretically possible) would yield the following transition matrix (which we will call the 'checking matrix' henceforth throughout this paper):

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{array}$$

This is ideal because we want the our keyboard to allow us to alternate hands as much as possible, which means RL and LR should be closer to 1 and RR and LL inversely should be closer to 0 since we don't want to type consecutive letters with the same hand. Throughout our paper you will also see us comparing these checking matrices, especially against QWERTY's checking matrix because our main goal is to exceed the performance of the QWERTY keyboard in our alternating hand metric. We want to see the better keyboards

have higher RL and LR percentages relative to the suspected worse keyboards, as well as lower RR and LL percentages.

4 Solutions

4.1 Overview

In our solution, we need a source text to first develop our transition matrix from a state of one character to the next, as described in Section 3.2 and Section 3.4. In this main solution, we use the modern literature text sample to accomplish this. The code runs through our list of modern literature texts and creates the transition matrix. This transition matrix is run through the corresponding keyboard creating methods listed below. The transition matrices are not printed in this paper but can be obtained through running the appropriate code listed in the appendix. After each keyboard is created, we can then create a checking matrix as described in Section 3.5. Each of the checking matrices are displayed below for their respective texts and keyboard layouts.

4.1.1 Terminology

RL denotes the probability of going from the right-hand state to the left-hand state.

LR denotes the probability of going from the left-hand state to the right-hand state.

QWERTYRL denotes QWERTY's RL.

QWERTYLR denotes QWERTY's LR.

RL Efficiency with QWERTYRL denotes the efficiency of a type of keyboard configuration for RL compare to QWERTY, where this values is larger or equal -1 and smaller or equal 1

LR Efficiency with QWERTYLR denotes the efficiency of a type of keyboard configuration for LR compare to QWERTY, where this values is larger or equal -1 and smaller or equal 1

For both RL and LR Efficiency, the result is simply the difference between the RL and LR probabilities of a particular keyboard and QWERTY.

4.2 Modern Literature

4.2.1 QWERTY keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.857 & 0.143 \\ 0.597 & 0.402 \end{pmatrix} \end{array}$$

L side character = A,S,D,F,G,Z,X,C,V,B,E,Q,R,T,W
R side character = Y,U,I,O,P,H,J,K,L,N,M

4.2.2 MANUAL PLACEMENT keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.827 & 0.173 \\ 0.756 & 0.244 \end{pmatrix} \end{array}$$

L side character = T,E,A,I,O,D,G,S,P,K,Q,X,Z
R side character = H,R,U,N,F,V,L,M,B,C,W,J,Y

Comparison with QWERTY
RL Efficiency with QWERTYRL = 0.159
LR Efficiency with QWERTYLR = 0.031

4.2.3 GREEDY ALGORITHM keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.814 & 0.186 \\ 0.506 & 0.495 \end{pmatrix} \end{array}$$

L side character = A,D,R,U,H,S,L,B,C,F,M,Q,X
R side character = N,E,O,T,I,P,Y,J,K,W,G,V,Z
Comparison with QWERTY
RL Efficiency with QWERTYRL = -0.092
LR Efficiency with QWERTYLR = 0.043

4.2.4 RANDOM PLACEMENT keyboard

We randomly placed 13 letters on each side for 20 times, and take the average of these 20 checking matrices.

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.782 & 0.218 \\ 0.559 & 0.441 \end{pmatrix} \end{array}$$

Comparison with QWERTY
RL Efficiency with QWERTYRL = -0.038
LR Efficiency with QWERTYLR = 0.075

4.3 Charles Dickens

4.3.1 QWERTY keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.798 & 0.201 \\ 0.618 & 0.382 \end{pmatrix} \end{array}$$

L side character = A,S,D,F,G,Z,X,C,V,B,E,Q,R,T,W
R side character = Y,U,I,O,P,H,J,K,L,N,M

4.3.2 MANUAL PLACEMENT keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.761 & 0.239 \\ 0.780 & 0.220 \end{pmatrix} \end{array}$$

L side characters = T,E,A,I,O,D,G,S,P,K,Q,X,Z
R side characters = H,R,U,N,F,V,L,M,B,C,W,J,Y
Comparison with QWERTY
RL Efficiency with QWERTYRL = 0.161
LR Efficiency with QWERTYLR = 0.037

4.3.3 GREEDY ALGORTHM keyboard

$$\begin{array}{cc} & \begin{array}{cc} L & R \end{array} \\ \begin{array}{c} L \\ R \end{array} & \begin{pmatrix} 0.773 & 0.227 \\ 0.474 & 0.526 \end{pmatrix} \end{array}$$

L side characters = A, D, R, U, H, S, K, Y, J, G, P, V, X
R side characters = N, E, O, T, I, C, L, B, F, M, Q, W, Z
RL Efficiency with QWERTYRL = -0.144
LR Efficiency with QWERTYLR = 0.026

4.3.4 RANDOM PLACEMENT keyboard

We randomly placed 13 letters on each side for 20 times, and take the average of these 20 checking matrices.

$$\begin{array}{cc} & \begin{array}{cc} L & R \end{array} \\ \begin{array}{c} L \\ R \end{array} & \begin{pmatrix} 0.771 & 0.229 \\ 0.526 & 0.474 \end{pmatrix} \end{array}$$

Comparison with QWERTY
RL Efficiency with QWERTYRL =-0.092
LR Efficiency with QWERTYLR = 0.028

5 Commentary

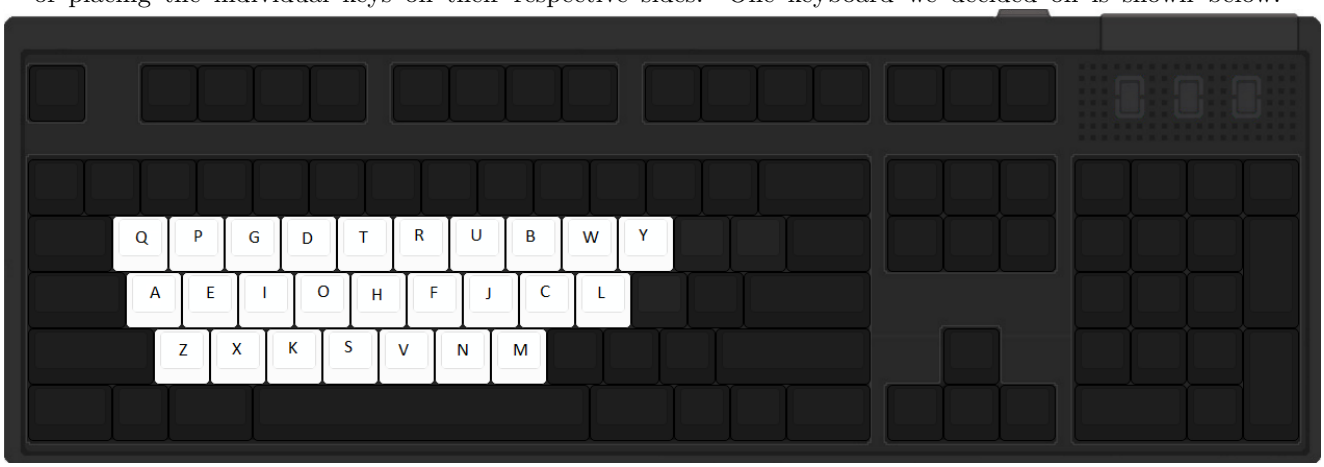
The goal of this model is to help us find a keyboard that can replace the QWERTY keyboard. We want to find what is known as our ALPHA keyboard, or our best solution keyboard. Looking at these results seems to suggest a clear winner. Overall, the keyboard that gives the best results is the manual placement keyboard. It gives us the best RL Efficiency with QWERTYRL and LR Efficiency with QWERTYLR, which tells us that we have a good alternation of hands with adjacent letters relative to the QWERTY standard keyboard. This is consistent across when we checked the keyboard's effectiveness against both modern texts and the Charles Dickens' texts. The RL efficiency is 16.1% better and LR efficiency is 3.7% better than QWERTY's respective efficiencies in the Dickens' texts, and then 15.9% and 3.1% better in RL and LR efficiency respectively for the modern texts.

Naturally the random keyboard didn't do so well, proving to have worse RL efficiency by around 7%, although slightly better LR efficiency than QWERTY at around 4% with our modern text samples. The checking matrix for the random result was averaged over twenty iterations of random layouts, so these results don't particular suggest QWERTY is much better than a random keyboard. Our greedy algorithm keyboard surprisingly fared similarly to the random keyboard in both Dickens and in the modern texts, performing just slightly better compared to random keyboard when comparing both RL and LR efficiencies with respect to the QWERTY keyboard. The respective efficiency results can be seen on Sections 4.2.4, 3.2.4, 4.2.3, and 3.2.3.

Another interesting aspect to note is that we very rarely encounter the situation where a keyboard results with a checking matrix with both RL and LR close to 1 and inversely LL and RR close to 0. This is true for almost all the keyboards in our result. A well known flaw with QWERTY is that over 70% of English words can be typed exclusively with the left hand. Consistently enough, we find that for QWERTY's LL efficiency tended to stay above 70% when analyzing both modern texts and Dickens' texts. Thus when we see that even though our other keyboards outperform QWERTY in LR and RL efficiency, they still actually tend to stay at a LL efficiency over 70%. This result may suggest that keyboards may have a certain limit to which we can truly design it to always alternate between left and right. There may simply be too many

combinations of adjacent letters that are valid in the English language to make such an optimization be greatly better than the QWERTY or even a randomly generated keyboard.

With the limitations of our model, we still can only decide whether to place a letter on the right or left side of the keyboard. Thus this gives us a lot of flexibility in making the final layout in terms of placing the individual keys on their respective sides. One keyboard we decided on is shown below:



6 Variations

6.1 LOL haaiii REDdit! AMA!

All the texts that we have analyzed so far have been written texts by mostly renowned authors. But the keyboard is used by much more than just formal writing. With cell phones and laptops and computers, a keyboard is very much prevalent and a necessity for all forms of communication. While writers tend to write formal and elegant English, the everyday user types with a plethora of slang, emojis, abbreviations, and well-placed expletives. The most representative "lol" in almost every online conversation caught our attention since it is all exclusively typed with the right hand on QWERTY. Thus we thought it is important to investigate text from a more casual setting, such as Reddit. We captured the response from a Reddit thread on the UDub subreddit where multiple people commented on the subject of computer science (text attached in the appendix under the name "reddit.txt").

We can look at how effective QWERTY and our keyboards are on this Reddit sample. Running the text through our model the same as we did in our solution, we found QWERTY to give us the following result matrix:

6.1.1 QWERTY Keyboard

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.797 & 0.203 \\ 0.565 & 0.435 \end{pmatrix} \end{array}$$

L side characters = A,S,D,F,G,Z,X,C,V,B,E,Q,R,T,W
R side characters = Y,U,I,O,P,H,J,K,L,N,M

6.1.2 ALPHA Keyboard

$$\begin{array}{cc} & \begin{array}{cc} L & R \end{array} \\ \begin{array}{c} L \\ R \end{array} & \begin{pmatrix} 0.761 & 0.239 \\ 0.783 & 0.217 \end{pmatrix} \end{array}$$

L side characters = T,E,A,I,O,D,G,S,P,K,Q,X,Z
R side characters = H,R,U,N,F,V,L,M,B,C,W,J,Y

Comparison with QWERTY
RL Efficiency with QWERTYRL = 0.218
LR Efficiency with QWERTYLR = 0.036

6.1.3 New GREEDY ALGORITHM Keyboard

$$\begin{array}{cc} & \begin{array}{cc} L & R \end{array} \\ \begin{array}{c} L \\ R \end{array} & \begin{pmatrix} 0.760 & 0.240 \\ 0.481 & 0.519 \end{pmatrix} \end{array}$$

L side characters: A, G, R, U, H, S, L, D, P, B, K, V, X
R side characters: N, E, O, T, I, C, Y, M, F, J, Q, W, Z

Comparison with QWERTY:
RL Efficiency with QWERTYRL = -0.084
LR Efficiency with QWERTYLR = 0.037

Relative to the QWERTY keyboard, our left to right state and right to left state for the ALPHA keyboard have an increase in 21.8% and 3.6% respectively, which bodes well for the usability of this keyboard. More importantly, we want to see if our ALPHA keyboard could work better for Reddit-like day-to-day conversations than for traditional written works like in our main solution. We compare the checking matrix for ALPHA's Reddit texts and the checking matrix for ALPHA's modern texts (data pulled from 4.1.2) and we find that there is actually a 6.6% increase in RL efficiency and 2.7% increase in LR efficiency. This suggests our ALPHA keyboard actually does slightly better with more casual English wording.

So interestingly our keyboard is actually rather effective for more conversational-type and informal English. It actually performs better for our formally written works compared to QWERTY, which suggests that our keyboard could have good potential to be used for multiple types of use.

We also use our greedy algorithm to create a new keyboard for the Reddit texts, but it seems to be that the greedy algorithm does not result in such an effective keyboard as it only outperforms the QWERTY with left-to-right key strokes by 3.7% and is less effective on right-to-left keys. Compared to the ALPHA keyboard, the RL and LR for ALPHA are higher by around 30% and 1% respectively, so clearly this particular created keyboard is not quite so effective.

6.2 Parle Français? Investigating French Texts

Our second variation is to investigate the use of French. Being one of the most widely spoken languages and having a rich history in texts, French is a great option to look into. We use three French works as our sources from different time periods and genres:

- *Les Miserables* by Victor Hugo (1862) [10]
- *Invasions des Sarrazins en France* by Joseph Toussaint Reinaud (1828) [16]
- *Entretiens* by Marie Lebert (2001) [12]

French essentially uses the same alphabet as English, with the exception of some added accents to vowels. In terms of typing however, we can assume that all letters with accents are the same as the letter without the accent (the user still presses the same key on the keyboard). While this is not the case on some traditional computer keyboards, it is implemented in this way on most mobile devices including the iPhone and Android, as well as on some virtual keyboards.

Now many French speaking countries have adopted an alternate keyboard to QWERTY called AZERTY. Barring some minor differences between countries, the general AZERTY keyboard layout is as follows:

A general rule of thumb for the left and right hand letters on the AZERTY keyboard are as follows:

Left: A, Z, E, R, Q, S, D, F, W, X, C, V, T
Right: P, O, I, U, Y, G, H, J, K, L, M, B, N

An initial assumption is that the AZERTY keyboard should perform better than QWERTY in French texts when we investigate our checking Markov matrix with states Left and Right. We want to maximize the probability of RL (state Right to state Left) and LR (state Left to state Right) to increase the likelihood that we type consecutive letters with opposite hands.

Using the method for creating our checking matrix as described in Section 3.5, the following are the matrices for the previously stated French texts using the following keyboards:

QWERTY:

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7976755291290551 & 0.20232447087094488 \\ 0.6089394591226691 & 0.39106054087733094 \end{pmatrix} \end{array}$$

AZERTY:

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7986147417578314 & 0.2013852582421686 \\ 0.5756655779307752 & 0.4243344220692248 \end{pmatrix} \end{array}$$

Thus we can see that QWERTY has a 60.89% chance of typing with the right hand followed by the left, while AZERTY has a 57.57%. Similarly QWERTY has a 20.23% chance of typing with the left hand followed by right, while AZERTY is at 20.14%, which is nearly identical. So our initial assumption is in fact false as we see QWERTY actually goes from right hand to left hand around 3% more than AZERTY. Part of this could

be due to the assumption that we forgo special accented characters as being separate keys. This would seem to be a big reason for the result, but our result may actually be more valid than it seems.

Many Francophiles have long complained about the shortcomings of the AZERTY keyboard, and in fact the French government's Culture and Research Ministry recently released a statement acknowledging the failures of the AZERTY keyboard to effectively type French [7]. They have launched an initiative to begin looking into developing a new alternative to AZERTY by the summer of 2016. The AZERTY keyboard appears to be no more effective that QWERTY in typing French in either our investigation or in the opinions of the general French-speaking populace.

Now we also have the checking matrix for ALPHA keyboard from our main model solution as follows:

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7738944562683319 & 0.2261055437316681 \\ 0.25310523177793326 & 0.2374822048224931 \end{pmatrix} \end{array}$$

Finding the difference between the AZERTY and our ALPHA keyboard for the RL and LR efficiency percentages gives us -30.25% and 2.47% respectively. So we can see our keyboard performs rather terribly going from right hand to left hand in comparison to AZERTY, while being at least somewhat more similar going from left to right. But that huge RL efficiency isn't particularly appealing. Our ALPHA keyboard really isn't going to cut it.

We decided to use the same methods to create our initial optimized English keyboard to create a new French keyboard. By following the same method outlined in our solution in Section 3.4.2 using our greedy algorithm (the transition matrix is not printed here for the sake of saving paper but can be found by running our French client code in the appendix), we came up with a keyboard with the following keys assigned to each hand:

Left: A, T, S, N, U, M, L, C, Y, J, G, K, X
Right: I, E, O, D, R, P, Q, H, B, F, W, V, Z

Our resulting checking matrix is:

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7573151677518084 & 0.24268483224819154 \\ 0.5797534353158971 & 0.420246564684103 \end{pmatrix} \end{array}$$

Conveniently, we find that this keyboard gives us 0.4% higher chance of RL and 4% higher chance of LR. This improvement is not massive, but shows that we can in fact use our model to find better keyboards for multiple different languages and texts.

6.3 *Fantastic Authors and Where to Find Them* - Let's Find J.K. Rowling

Most people have a favourite author. There could be many reasons one enjoys the work of an author, ranging from the subject theme and content to the writing style they employ. Writing style, in particular, happens to be something that is rather unique to each writer. Could it be possible to scan a piece of writing and find out who the author is? It appears to linguists that it should be the case.

One particularly fascinating case involved J.K. Rowling, the author of the widely popular Harry Potter series. She secretly released the novel The Cuckoo's Calling under the pseudonym Robert Galbraith in 2013. Sales were initially slow, but when it was discovered that Rowling was the author, the novel shot up to the top of Amazon best-seller's list [22]. Before Rowling admitted to being the author, a news reporter received a tip that Rowling could be the potential author. Wanting to make sure, the reporter enlisted a Duquesne University professor who specialised in identifying authors in unknown works based off technology analyzing

word frequency, word length, word adjacency, and letter clusters. His results strongly supported the notion that Rowling was the author, to which Rowling admitted to shortly after the results were published [19].

This detective work brings up an interesting scenario. Could we create a keyboard for J.K. Rowling's works, and then use that keyboard to help us discover if other works are written by her? Since we know she writes with a similar style across her novels, we could use our model to try to see if we can distinguish her other works to works of different authors using our "Rowling" keyboard with the same methods as above.

Conveniently, Rowling has a good amount of written material to use as our source to create the keyboard. Copyright laws allow us to copy a chapter or 10% of the text (whichever it greater) from each of her seven Harry Potter novels to use for our investigation. Using Amazon, we can get a preview chapter of each novel and then copy the text for use legally.

Plugging the combination of the texts into our model as specified in our main solution and using our algorithm to create a keyboard, we get the keyboard (which we will call the "Rowling keyboard") as follows:

Left: A, G, E, O, T, S, Y, D, P, J, K, Q, X
Right: N, H, R, U, I, L, W, M, B, C, F, V, Z

Our resulting checking matrix is as follows:

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7702506306135745 & 0.2297493693864255 \\ 0.6642444072439507 & 0.3357555927560493 \end{pmatrix} \end{array}$$

This matrix will be our baseline to compare her other work and other authors works. Now, Rowling has another novel called The Casual Vacancy along with The Cuckoo's Nest. Similar to the method in our main solution (section 3.4-3.6), we can use our Rowling keyboard layout and model a Markov chain matrix of how her adjacent letters move from hand to hand for each text. We get the following resulting matrix:

The Cuckoo's Calling by J.K. Rowling [17]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7571362205130853 & 0.2428637794869148 \\ 0.6647283406754773 & 0.33527165932452274 \end{pmatrix} \end{array}$$

The Casual Vacancy by J.K. Rowling [18]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7577867599605992 & 0.24221324003940084 \\ 0.6645009088448003 & 0.33549909115519977 \end{pmatrix} \end{array}$$

Catch-22 by Joseph Heller [9]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7577346617724174 & 0.2422653382275826 \\ 0.6680597974602154 & 0.3319402025397846 \end{pmatrix} \end{array}$$

The DaVinci Code by Dan Brown [2]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.748625598050578 & 0.251374401949422 \\ 0.6566213643227627 & 0.3433786356772372 \end{pmatrix} \end{array}$$

The Shining by Stephen King [11]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.7709191699752791 & 0.22908083002472096 \\ 0.6685864836713437 & 0.33141351632865634 \end{pmatrix} \end{array}$$

To Kill a Mockingbird by Harper Lee [13]

$$\begin{array}{c} \\ L \\ R \end{array} \begin{array}{c} \begin{array}{cc} L & R \end{array} \\ \begin{pmatrix} 0.22866011381272633 & 0.7713398861872737 \\ 0.6685864836713437 & 0.33453485696755886 \end{pmatrix} \end{array}$$

Using Harry Potter excerpts as our standard, we can find the average of all the absolute value differences between each novel and Harry Potter of each first left/right state to the second left/right state. This is displayed in the table below:

	RR	RL	LR	LL	Average State Change	Smallest State Change
The Casual Vacancy	0.0002565016008	0.0002565016008	0.01246387065	0.01246387065	0.006360186125	0.0002565016008
The Cuckoo's Calling	0.0004839334315	0.0004839334315	0.0131144101	0.0131144101	0.006799171766	0.0004839334315
Catch-22	0.003815390216	0.003815390216	0.01251596884	0.01251596884	0.008165679528	0.003815390216
The DaVinci Code	0.004342076427	0.004342076427	0.0006685393617	0.0006685393617	0.002505307894	0.0006685393617
The Shining	0.007623042921	0.007623042921	0.02162503256	0.02162503256	0.01462403774	0.007623042921
To Kill a Mockingbird	0.001220735788	0.001220735788	0.001089255574	0.001089255574	0.001154995681	0.001089255574

Now we see that by our metric of averaging the absolute distance of each state to another state, we find that both of Rowling's novels are very close to the matrix we get from her Harry Potter novels. *The Cuckoo's Calling* has an average difference of 0.64% while *The Casual Vacancy* has an average difference of 0.68%. However, as we can see above, *To Kill a Mockingbird* has the smallest difference, at a minuscule 0.1% difference from Rowling's *Harry Potter* excerpt. So using this difference metric really doesn't tell us much.

However, if we look at the smallest difference in probability from a state to another state for each novel, we do in fact find that Rowling's two novels are in both most similar to *Harry Potter* with a 0.02% and 0.04% difference when the first letter of any two adjacent letters is typed with the right hand. Of course, given the somewhat small sample size, copyright restrictions, and limitations of how strongly the keyboard we created for Rowling actually fit, we don't know for sure how accurate this model is other than it worked given this set of restrictions. Given that most linguists require much more detail from a person's work to identify them, this model nevertheless shows it can be used as a potential step to help solidify writer identification.

7 Conclusion

We tried various methods to improve the current QWERTY layout by doing textual analysis using Markov chains. Overall, the manual placement keyboard outperforms other keyboards and became our ALPHA keyboard. It has moderately better LR and RL efficiencies than QWERTY keyboard throughout the earlier, modern literature, and the more casual online thread we chose.

However, since the keyboard is manually placed, its performance may not be fully optimized. That is one thing we might need to further investigate into. And there might also be a better algorithm to efficiently assign letters than our greedy algorithm mentioned above. Another improvement we can make in the future is to rank the efficiency of each finger and take into account of ergonomic feasibility. And by doing so we might be able to create a keyboard that is more realistic for the everyday user.

In addition, even though our ALPHA keyboard has a better performance, the learning curve could be steep enough to jeopardize the excess efficiency it produces. But overall this project demonstrates that it is possible to redesign the QWERTY keyboard and get improved performance.

8 Appendix

8.1 Java Code

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.*;
import java.text.DecimalFormat;

public class CharacterProcessorClientRandomTest {
  public static final int N = 26; // number of letters in alphabet
  public static int count;
   public static int countCheckTop;
   public static int countCheckBot;

  public static void main(String[] args) throws FileNotFoundException {
    int[][] charMatrix = new int[N][N]; // our character matrix,
    count = 0;
    countCheckTop = 0;
    countCheckBot = 0;
    List<String> files = new LinkedList<String>();

  //  files.add("adventuresofhuckfinn.txt");
//      files.add("Prime-Difference-by-Alan-Edward-Nourse.txt");
  //  files.add("In-Bad-Company-and-other-stories-by-Rolf-Boldrewood.txt");
     files.add("Break-a-Leg-by-Jim-Harmon.txt");
//      files.add("A-Dog-Day-by-Walter-Emanuel.txt");
    // ADD FILES in format files.add('filename.txt')


     // FRENCH FILES
     //files.add("lesmis.txt");
    // files.add("Invasions-des-Sarrazins-en-France.txt");


    for (String i: files) {
      charMatrix = addCounts(charMatrix, i);
    }

    System.out.println(Arrays.deepToString(charMatrix));

      // Probability
      double[][] probability = new double[N][N];
      probability =  probabilityFinder(charMatrix, probability);

      // priority queue
      PriorityQueue<LetterCombination> orderedProb = new PriorityQueue<
          LetterCombination>(new LetterCombination());
      for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
          char i_letter = (char) (i + 97);
          char j_letter = (char) (j + 97);
          LetterCombination letterCombination = new LetterCombination(i_letter,
              j_letter, probability[i][j]);;
          orderedProb.add(letterCombination);
        }
      }

      // print priority queue
```

```
    while (orderedProb.peek() != null && orderedProb.peek().getPercentage() > 0) {
        String letterCombination = "";
        if (orderedProb.peek().getPercentage() > 0) {
            letterCombination = orderedProb.poll().toString();
            //System.out.println(letterCombination);
        }
    }



    //declare L or R for QWERTY

    System.out.println("==========");
    System.out.println("QWERTY");
    System.out.println("==========");
    char [] QWERTYleft =  {'a','s','d','f','g','z','x','c','v','b','e','q','r','t'
        ,'w'};
    char [] QWERTYright = {'y','u','i','o','p','h','j','k','l','n','m'};

    Map<Character, String> trackLR = new HashMap<Character, String>();
    for (char a : QWERTYleft) {
        trackLR.put(a,"L");
    }
    for (char a : QWERTYright) {
        trackLR.put(a,"R");
    }
    double[][]QWERTYLR = new double[2][2];
    for (String i: files) {
    QWERTYLR = sortLR(trackLR, i);
}

System.out.println(Arrays.deepToString(QWERTYLR));
    QWERTYLR[0][0] = QWERTYLR[0][0]/countCheckTop;
    QWERTYLR[0][1] = QWERTYLR[0][1]/countCheckTop;
    QWERTYLR[1][0] = QWERTYLR[1][0]/countCheckBot;
    QWERTYLR[1][1] = QWERTYLR[1][1]/countCheckBot;
    System.out.println(Arrays.deepToString(QWERTYLR));




    //declare L or R

    System.out.println("==========");
    System.out.println("MANUAL BY HAND");
    System.out.println("==========");
    char [] OURleft =  {'t','e','a','i','o','d','g','s','p','k','q','x','z'};
    char [] OURright = {'h','r','u','n','f','v','l','m','b','c','w','j','y'};

    Map<Character, String> trackOURLR = new HashMap<Character, String>();
    for (char a : OURleft) {
        trackOURLR.put(a,"L");
    }
    for (char a : OURright) {
        trackOURLR.put(a,"R");
    }
    double[][]OURLR = new double[2][2];
    for (String i: files) {
```

```
  OURLR = sortLR(trackOURLR, i);
}
//System.out.println(Arrays.deepToString(OURLR));
  OURLR[0][0] = OURLR[0][0]/countCheckTop;
  OURLR[0][1] = OURLR[0][1]/countCheckTop;
  OURLR[1][0] = OURLR[1][0]/countCheckBot;
  OURLR[1][1] = OURLR[1][1]/countCheckBot;
  System.out.println(Arrays.deepToString(OURLR));

  double compareQWERTYwithOURRL =OURLR[0][1] - QWERTYLR[0][1];
  double compareQWERTYwithOURLR =OURLR[1][0] - QWERTYLR[1][0];
  System.out.println("RL = " + compareQWERTYwithOURRL);
  System.out.println("LR = " + compareQWERTYwithOURLR);




  countCheckTop = 0;
countCheckBot = 0;

System.out.println("==========");
    System.out.println("USING COMPUTER ALGORITHM");
    System.out.println("==========");
   // A different LR configuration
LeftRightFinder lr = new LeftRightFinder(probability, N);
LinkedList<LinkedList<Character>> result = lr.createLR();

LinkedList<Character> left = result.get(0); // left letters
LinkedList<Character> right = result.get(1); // right letters

System.out.println(left.toString());
System.out.println(right.toString());

 Map<Character, String> trackCompLR = new HashMap<Character, String>();
    for (char a : left) {
       trackCompLR.put(a,"L");
    }
    for (char a : right) {
       trackCompLR.put(a,"R");
    }
    double[][]COMPLR = new double[2][2];
    for (String i: files) {
   COMPLR = sortLR(trackCompLR, i);
  }
  //System.out.println(Arrays.deepToString(OURLR));
  COMPLR[0][0] = COMPLR[0][0]/countCheckTop;
  COMPLR[0][1] = COMPLR[0][1]/countCheckTop;
  COMPLR[1][0] = COMPLR[1][0]/countCheckBot;
  COMPLR[1][1] = COMPLR[1][1]/countCheckBot;
   System.out.println(Arrays.deepToString(COMPLR));

   double compareQWERTYwithCOMPRL2 =COMPLR[0][1] - QWERTYLR[0][1];
    double compareQWERTYwithCOMPLR2 =COMPLR[1][0] - QWERTYLR[1][0];
    System.out.println("RL = " + compareQWERTYwithCOMPRL2);
    System.out.println("LR = " + compareQWERTYwithCOMPLR2);
```

```
        System.out.println("==========================");
        System.out.println("MANUAL BY HAND 2");
        System.out.println("==========================");
       countCheckTop = 0;
      countCheckBot = 0;
      char [] OURleft2 =  {'a', 'd', 'r', 'u', 'h', 's', 'l', 'b', 'c', 'f', 'm', 'q
          ', 'x'};
      char [] OURright2 = {'n', 'e', 'o', 't', 'i', 'p', 'y', 'k', 'j', 'w', 'g', 'v
          ', 'z'};
      Map<Character, String> trackOURLR2 = new HashMap<Character, String>();
      for (char a : OURleft2) {
         trackOURLR2.put(a,"L");
      }
      for (char a : OURright2) {
         trackOURLR2.put(a,"R");
      }
      double[][]OURLR2 = new double[2][2];
      for (String i: files) {
      OURLR2 = sortLR(trackOURLR2, i);
    }
    //System.out.println(Arrays.deepToString(OURLR2));
      OURLR2[0][0] = OURLR2[0][0]/countCheckTop;
      OURLR2[0][1] = OURLR2[0][1]/countCheckTop;
      OURLR2[1][0] = OURLR2[1][0]/countCheckBot;
      OURLR2[1][1] = OURLR2[1][1]/countCheckBot;
      System.out.println(Arrays.deepToString(OURLR2));

      double compareQWERTYwithOURRL2 =OURLR2[0][1] - QWERTYLR[0][1];
      double compareQWERTYwithOURLR2 =OURLR2[1][0] - QWERTYLR[1][0];
      System.out.println("RL = " + compareQWERTYwithOURRL2);
      System.out.println("LR = " + compareQWERTYwithOURLR2);



      System.out.println("==========================");
      System.out.println("RANDOM KEYBOARD");
      System.out.println("==========================");
      countCheckTop = 0;
    countCheckBot = 0;
      // Random
      Random rand = new Random();
//
      char [] randCLeft = new char [N/2];
      char [] randCRight = new char [N/2];
      int [][]dummyRand = new int [2][2];
      boolean flag = true;
      for (int i = 0; i < count; i++) {
         for (int j = 0; j < randCLeft.length; j++) {
            int randomInt = rand.nextInt(122 - 97+ 1)+97;
            flag = true;
            for (int k = 0; k < randCLeft.length; k++) {
               if (randCLeft [k] == randomInt) {
                  flag = false;
                  j--;
               }
            }
            if (flag) {
               randCLeft [j] = (char) randomInt;
            }
```

```
            }
        }

        flag = true;
        for (int i = 0; i < count; i++) {
            for (int j = 0; j < randCRight.length; j++) {
                int randomInt = rand.nextInt(122 - 97+ 1)+97;
                flag = true;
                for (int k = 0; k < randCRight.length; k++) {
                    for (int r = 0; r < randCLeft.length; r++) {
                        if (randCRight [k] == randomInt || randomInt == randCLeft[r]) {
                            flag = false;

                        }
                    }
                }
                if (flag) {
                    randCRight [j] = (char) randomInt;
                }
            }
        }
        System.out.println(randCLeft);
        System.out.println(randCRight); ;
        Map<Character, String> trackOURLRRand = new HashMap<Character, String>();
        for (char a : randCLeft) {
            trackOURLRRand.put(a,"L");
        }
        for (char a : randCRight) {
            trackOURLRRand.put(a,"R");
        }
        double[][] randCheck = new double[2][2];
        for (String i: files) {
        randCheck = sortLR(trackOURLRRand, i);
    }
    System.out.println(Arrays.deepToString(randCheck));
        randCheck[0][0] = randCheck[0][0]/countCheckTop;
        randCheck[0][1] = randCheck[0][1]/countCheckTop;
        randCheck[1][0] = randCheck[1][0]/countCheckBot;
        randCheck[1][1] = randCheck[1][1]/countCheckBot;
        System.out.println(Arrays.deepToString(randCheck));

        double compareQWERTYwithOURRLRand =randCheck[0][1] - QWERTYLR[0][1];
        double compareQWERTYwithOURLRRand =randCheck[1][0] - QWERTYLR[1][0];
        System.out.println("RL = " + compareQWERTYwithOURRLRand);
        System.out.println("LR = " + compareQWERTYwithOURLRRand);

//        System.out.println(Arrays.deepToString(probability));
    }

    // Create checking Matrix
    // Create checking Matrix
        private static double[][] sortLR (Map<Character, String> trackLR, String file)
            throws FileNotFoundException {
            countCheckTop = 0;
            countCheckBot = 0;
            double[][] checkMatrix = new double [2][2];
            Scanner input = new Scanner(new File(file));
        String cumulative = "";
        while(input.hasNextLine()) {
```

```
      String stringLine = input.nextLine();
      cumulative += stringLine.toLowerCase();
    }
    boolean isLeft;
      boolean isRight;
    char[] charList = cumulative.toCharArray();
    for (int i = 0; i < charList.length - 1; i++) {
         char first = charList[i];
         char second = charList[i + 1];
         String firstC = trackLR.get(first);
         String secondC = trackLR.get(second);
         if (firstC == "R" && secondC == "R") {
            checkMatrix[0][0]++;
            countCheckTop++;
         } else if (firstC == "R" && secondC == "L") {
            checkMatrix[0][1]++;
            countCheckTop++;
         } else if(firstC == "L" && secondC == "R") {
            checkMatrix[1][0]++;
            countCheckBot++;
         } else  {
            checkMatrix[1][1]++;
            countCheckBot++;
         }
    }
      System.out.println(countCheckTop);
      System.out.println(countCheckBot);
      return checkMatrix;
   }

// create matrix
private static int[][] addCounts(int[][] charMatrix, String file) throws
    FileNotFoundException {
  int newCharMatrix[][] = charMatrix;

  Scanner input = new Scanner(new File(file));

  String cumulative = "";
  while(input.hasNextLine()) {
    String stringLine = input.nextLine();
    cumulative += stringLine.toLowerCase();
  }

  char[] charList = cumulative.toCharArray();


  for (int i = 0; i < charList.length - 1; i++) {
       char first = charList[i];
    if (isValidCharacter(first)) {
      char second = charList[i+1];
      if (isValidCharacter(second)) {
        int val1 = first - 97;
        int val2 = second - 97;
        newCharMatrix[val1][val2]++;
             //System.out.println(val1 + " " + val2); //to print the index of the
                 2D array
             count++;
        first = second;
      }
```

```
      }
    }

    return newCharMatrix;
  }

  // check if character is a letter
  private static boolean isValidCharacter(char a) {
    int value = (int) a;
    return (value >= 97 && value <= 122);
  }

   //Wen Lin 's modification to find the probablity
   private static double[][] probabilityFinder(int[][] charMatrix, double[][]
       probability) {
    double newProbability[][] = probability;
      DecimalFormat dec = new DecimalFormat("#.000000");
      for (int i = 0; i < N; i++) {
         for (int j = 0; j < N; j++) {
            int elementCounts = charMatrix[i][j];
            double elementProbability = 0.00001d;
            elementProbability = (double) elementCounts/ (double) count;
            newProbability[i][j] = Double.valueOf(dec.format(elementProbability));
            if (elementProbability > 0) {
               char first = (char)(i + 97);
               char second = (char)(j + 97);
               //System.out.println(first + " "+ second+ " = " + elementProbability
                   + " " + i + " " + j  );
            }
         }
      }
      return newProbability;
   }
}

// NEW: LetterCombination class to hold contents in PriorityQueue
class LetterCombination implements Comparator<LetterCombination> {
  private char firstLetter;
  private char secondLetter;
  private double percentage;

  // default constructor
  public LetterCombination() {
    firstLetter = '␣';
    secondLetter = '␣';
    percentage = 0;
  }

  public LetterCombination(char first, char second, double perc) {
    firstLetter = first;
    secondLetter = second;
    percentage = perc;
  }

  public char getFirstLetter() {
    return firstLetter;
  }

  public char getSecondLetter() {
```

```
    return secondLetter;
  }

  public double getPercentage() {
    return percentage;
  }

  public String toString() {
    return "" + firstLetter + "" + secondLetter + "␣" + percentage;
  }

  @Override
  public int compare(LetterCombination o1, LetterCombination o2) {
    double o1_perc = o1.getPercentage();
    double o2_perc = o2.getPercentage();
    if (o1_perc == o2_perc) {
      return 0;
    } else if (o1_perc > o2_perc) {
      return -1;
    } else {
      return 1;
    }
  }
}
```

8.2 Reddit text

```
Probably both (except not dumb), along with any other number of reasons.
I spent my first two years trying and failing to get into the Computer Science major and eventually I had to settle
      with Mathematics even though I had taken very few major-required math classes, so I now I have less time to
     take the same amount of classes. Personally, that's why I need an extra quarter to graduate.

Why do people take more than 4 years to graduate (besides majors where it might be needed, such as engineering and
     accounting)? Like I don't get it. Are they just delaying the job search? Do they like UW? I can't imagine
     willingly staying at UW for more than even 3 years, and my friends wanna gtfo asap as well.
A few reasons off the top of my head:
1.       Applying to a major (once or multiple times), being rejected, having to choose something else, and being
     behind in prereqs for your backup major.
2.       Studying abroad can delay your graduation date because not at all study abroad fulfill strict requirements
     of a particular major. Studying abroad is still a worthwhile experience.
3.       "A dumb double major" as you so eloquently put it.
I think it's extremely ignorant to assume everyone has a negative view of UW. I am currently in my 5th year doing a
      double major and I have absolutely no regrets.
I haven't been at UW for 4 years, but I have been in college for over 4 years (went to community college for first
     2 years then transferred).
Some of us change majors and hence take more than 4 years. I was pursuing a different major but then changed to a
     math major. This has caused me to require being in college for 5 years in order to graduate.
I don't really see the big incentive to rush through college when the job market is shit anyways. Furthermore I'm
     not racking up any debt either.
Another point is that I'd rather take my time and pick the right major as opposed to rushing through college and
     then realizing afterwards that I picked the wrong major. You're only in college for a few years whereas you'll
      be working for 40+ years after college. Doesn't make much sense to me to rush through college.
One last reason why I have taken long is because I have a very grim view of the future. My focus is on making
     decisions that are good for me in the long term and not necessarily in the short term.
The future entails growing income inequality, Social Security and Medicare are growing out of control, special
     interests and corporations run our government, mass atomization is close and will happen in the next 10-20
     years (my brother studies artificial intelligence, if you think only blue collar jobs are going to be atomized
      you're wrong. In fact there's more incentive to atomize expensive white collar jobs). We also live in a
     culture that values hedonism and does not value thoughtfulness, a result of this is an extremely ignorant
     voting base. If you look at polling data fewer and fewer people even read anymore and this includes the
     college graduate population.
All of this is a very toxic mix for the future, it looks very grim and I am not foolish enough to rely on people
     being competent enough to fix it. The government has no incentive to fix these problems because the government
      is controlled by special interests who want things to go this way and as I've pointed out the population
     simply put is ignorant and anti-intellectual, they lack the intellectual tools to combat the sophisticated
     world we live in.
This may seem a bit far out there but this line of reasoning is a significant part of why I'm really really making
     sure what I major in makes sense for the future. Math is only going to become more relevant in the future.
     Math is flexible and allows you to do a ton of things with it. I could go into software engineering, or
     finance, or engineering, or economics, or operations research, etc, etc.
Even if it took someone up till the age of 30 to graduate working for 40 years would mean working until they're 70.
      Lots of people work into their 70's so actually my claim of working 40+ years is completely accurate.
Furthermore as life expectancy continues to grow people not only will be able to work for more years, but it will
     be a necessity due to both growing income inequality and the fact that they will need more money due to their
     lives lasting longer.
I dont think the future will be THAT bad.
Yeah well I'm sure you've put a lot of thought into the future because clearly you're a very thoughtful person who
     puts a lot of effort into what they believe.
Yeah well good luck with that.
Also it would be nice if you wrote posts that weren't so incoherent. How'd you get into UW anyways? Hopefully you
     aren't this stupid in real life and are just trolling cause you're bored and have nothing better to do.
holy shit you actually believe ppl will work that long? have you heard of unemployment? basic income? welfare?
     machines doing all the work for us? holy fuck no wonder the right is seen as a bunch of fucking idiots. HOLY
     SHIT you are FUCKING DUMB, like legit dumbest guy on this subreddit, wtf? dude... people are not gonna work
     that long ok.. omg... fuck this guy
```

dude im gonna be completely honest with you... you should reevaluate your thoughts on what society in the future will look like. greater income inequality does NOT mean people will work that long... HOLY SHIT i am triggered af, gimmie a safe space

My wife and I would like to tour the campus when the cherry trees are most beautiful. Are they ready yet?

3rd and 4th weeks of March tends to be their peak time, but it could be different because of all the rain and warm weather this year. I'm sure someone will also post in this sub when they finally bloom.

I think that being out of state puts you at an advantage. Your tuition is higher therefore you have more value to the school, in some respects. Getting into the design program was the hardest thing I've ever done in my life but the benefits are tremendous. It's probably one of the most competitive majors on campus. Also don't make up your mind about IxD just yet lol. A ton of my friended ended up on tracks they didn't think they were going to :)

Hi! did you feel like it was hard getting in as an OOS? My high school is one of the top in MA so my gpa is definitely a little lower than it would be somewhere else. Was it hard to get into Industrial Design? I'm set on Interaction Design, but I've read online that it can be competitive to get into at UW.

Why do people take more than 4 years to graduate (besides majors where it might be needed, such as engineering and accounting)? Like I don't get it. Are they just delaying the job search? Do they like UW? I can't imagine willingly staying at UW for more than even 3 years, and my friends wanna gtfo asap as well..

Hi everyone. I'm currently a math major at the University of Illinois (Urbana). I'm in my third year right now, and I am very interested in the graduate program in mathematics at UW. I did scan through the admission process at the math department homepage. I just wonder what are the most essential factors when applying to the grad program at UW? Thanks a lot

I'm applying for residency next year and was wondering how it went for others and you guys have any tips. For proof I was going to submit rent info, transcripts, and work stubs and wanted to see if anyone has suggestions for other proof. Thanks!

I just got accepted to the MSE (master's science engineering) program at Udub. Since I'm a master's student, I didn't get any RA or TA funds. Does anybody have any experience with the program? It's not the MSME, which is for ME undergrads (I have a physics degree), is there a non-negligible difference? What opportunities are there other than RA or TA to pay for it, i.e. internships or work in the department? Any experience or helpful advice would be great, I'm trying to decide if I want to accept the seat.

I am a grad student (24F) and I am trying to find some roomies. Anyone looking for housing too or have a empty place in September?

Here is my background:

- GPA overall/major/upper-division major: 3.45/3.21/2.97
- Took 5 years to complete my undergrad at a state school
- Retook some courses
- Withdrew and failed some courses during my junior year
- Honor's College member
- 3 terms of Honor Roll
- Pi Mu Epsilon Math Honors Society Member
- Conducting research with math professor in Applied Mathematics for 1 year
- Presenting at 1 poster fair for my school's Honor's College in May 2014
- Writing my undergrad thesis and expecting to defend in Sept 2014
- Intend to take the GRE and math subject exam Oct 2014
- Apply winter '14/'15 to UW

What are my chances for funding for an MS?

Commencing pasta from 3/24/2015 post

////////////////////////////////////

//// UW CSE FAQ ////

////////////////////////////////////

BEFORE APPLYING

How strong is the UW CSE program?

UW CS is generally regarded to be a top 10 program in the nation. It is considered a tier below: Berkeley, MIT, CMU, and Stanford. It is considered comparable to: UIUC, UT: Austin, Waterloo, Princeton, Cornell, UMich, etc.

How can I get accepted to the program right out of high school?

For the Direct Admit program, UW CSE looks mainly for in-state students with high GPAs (3.8+), high test scores (SAT 2200+), and a track record of excelling in advanced courses. This is mainly to dissuade students who may be going to MIT, CMU, and other top tier CS programs. This page provides more information on CSE DAs. These admissions go out a few weeks after general UW admissions and are sent to students who put Computer Science as their intended major.

Ok, if I'm not accepted through the DA program, what does it take to get it through upper level admission?

A few definitive resources on how competitive upper level admissions is:

A broad picture of the number of students intending to apply to CSE compared to other engineering majors.

Some historical admissions statistics including ethnicity, gender, GPA, and % offers. This is almost entirely what you should base your chances on unless you have extenuating circumstances that would enhance your application.

An analysis of intro CS grades: how to do well and what it takes to be at the top of the class.

The most important classes to do well (in my opinion) are ranked as such from most important to less important: CSE 3XX (if you petition in), CSE 143(X), CSE 142, MATH 126, MATH 12X, PHYS 12X, CHEM, BIO, ENGL 111. Having AP Science courses instead of UW science courses doesn't hurt your application if you have very high scores in all your other classes. Doing well in the science courses however, is an additional vote of confidence (esp. physics).

Should I go to UW if I am dead set on doing UW CSE?

It depends. If you know you can perform at the levels outlined above (roughly between DA stats and Upper level stats) and work hard, yes. It's a high bar, but admissions is by no means a crapshoot! Admissions is quite consistent, so it depends on your comfort level in the required courses and subjects. If you are unsure/are borderline, please consider the other majors. UW is a great school outside of the CS program. If you refuse to try another major, by all means, go to a school you think fits you best.

APPLICATION

What criteria do admissions look at?

By far, admissions looks at your performance in the important classes outlined above. The essay represents a small buffer that can sway your decision either way UNLESS you show some extenuating hardship or special reason you should be admitted. This works in reverse too - being condescending, ignorant, or disrespectful is a red flag that CAN negate any 4.0 GPA. This has happened in the past.

IF REJECTED

What alternatives do I have if I want to go into software development?

Common backup programs for UW CSE students include Informatics, HCDE, Math, and ACMS. Informatics is growing to be quite competitive as well, which is compounded by the students applying as a backup or alternative to CSE. These programs are all strong and can prepare you for a software engineering job out of school. Take a look at their undergraduate webpages.

What alternatives do I have if I want to go into a more hardware-focused computer engineering field?

The Electrical Engineering program at UW has a major concentration called Embedded Devices that has recently been changed to share course material with CSE. Check out the official PDF of this Digital Overhaul. If your interests lie in the more lower level programming and digital design, then EE is a great alternative.

Should I retake X class?

It depends on the class, the grade you got, and the grade you will get after retaking it. You have a huge advantage in terms of material familiarity when retaking, so you are expected to do very well if you choose this route. If you are unsure, please talk to an adviser. Generally, a high CSE 143 grade already overshadows a low CSE 142 grade.

How else can I prove myself?

Take higher level courses instead of retaking classes if you can. Try and petition in to upper level CS classes. Note that this is mostly reserved for students who have been rejected by the department already. There are limited spots for classes such as 311, 332, and 351, but talking to an adviser or meeting in person with the professor teaching the class could work. Otherwise, take courses such as MATH 307 or 308 and do well in them.

MISC

What major pathways are available to a CSE major?

Check out this resource on what options exist for CSE majors in terms of specialization. These include bioinformatics, graphics, gaming, databases, ML, NLP, robotics, etc. There are also some great joint programs between CSE and say the Business school (software entrepreneurship) or the Linguistics department (computational linguistics) for example.

What about (X thing not in the FAQ)?

Post a comment in this thread. If the question is good and the answers are good, it'll go into the FAQ. Take all answers with a grain of salt even if it has a lot of upvotes. For a final vote of confidence, email or set up a meeting with a CS adviser. They can be vague sometimes, but that's just so they do not spread unnecessary misinformation to students.

CONCLUSION

I hope this helps everyone in their decision making processes when considering UW CSE! I can vouch that it is a fantastic program and UW as a whole has great opportunities for everyone. Please let me know if there are any improvements for this guide, other important resources to look at, or anything you think is plain wrong. I want this subreddit to be a place for students to get a better sense of the community and standards here at UW.

Thanks for reading!

Is anyone in Prof Warner's math 328 class this quarter?

I would love to talk to anyone who has had him for 328.

What makes them all come here? Is there some common path theyre on from Portland to Vancouver? They're annoying as fuck to be honest. Threatening, travel in packs, and get all pissed when you dont give them free handouts. It's not like UW students have tons of money to throw at them so how is this a gathering place for so many of what I can only describe as "troublemaking hippie-grunge kids"?

Any grad school students in here? Any students who came from out of state? I am from CA and just got word that I have been offered admission to UW's Masters of Public Health Program. I have done a lot of research up to this point, but wanted to see if anyone on here is in the same, or similar, position. Thoughts on moving to Seattle? Best neighborhoods to live in? Any advice is much appreciated as UW is definitely my top choice.

mentioned that Seattle is expensive. Coming from California, you likely won't think so. I'm from the Bay Area and everything here is significantly cheaper.

As for neighborhoods, it's very difficult to recommend without you seeing them. I'd suggest visiting for a while and looking around. If that's not feasible, live in the UDistrict for a quarter or so and spend some time checking out the rest of Seattle.

Nice! I currently live in Sac. They do have a prospective student day that I will attend and take that opportunity to check out to surrounding neighborhoods. Thanks for the reply!

I'm a physics grad, so I don't know much about your program. Here's what I'll say: Seattle is an expensive place to live, but it's also great for music, sports, outdoorsy shit, you name it. As a student, you'll get a great deal on a U-Pass, which lets you on just about any bus/train in the city.

As for places to live, that depends on what you're looking for. UW is easily accessible by bus from most areas nearby, so you don't have to live in the U-District, which can get sketchy in places.

Former grad student in MSIM. I don't know your program all that well, but I can give some insight about living in the area as a grad student.

Living in Seattle compared to California will probably be cheaper, but still more expensive than if you lived in another part of the state.

I lived in Wedgwood while in school, close enough to be convenient, but far enough away to have less distractions. It might be a good place to check out.

Definitely take advantage of your u pass, even if you drive. Seattle has a decent transportation system and a less than great parking situation in most populated areas.

Depending on your interests, Seattle is a wonderful place to be. You like the outdoors? Head up to the mountain loop highway and go hiking. Go east to Snoqualmie for more mountains. Skiing, hiking, camping.

You like being indoors? That's cool too. Grad school is time consuming. Lots of great places on campus to use free wifi and get work done, or depending on your neighborhood, great coffee shops or book stores to be in.

I don't know much about your program, but I'd recommend checking our potential employers that you'd be interested in for internships, jobs etc. Seattle has a lot to offer for tech, medical etc.

I've finally done it guys. I've discovered why there are so many English and Art Majors. It's not because we don't have the ability to become engineers or doctors, it's because class registration could be better handled by a pack of rhesus monkeys. If I wanted to take courses online I wouldn't be at this school. Congrats UW, you've spawned another Art major

open up for a class I wanted to take. Last quarter my measles vaccine information magically disappeared from the database making me miss registration then also. You ever tasted a dandelion leaf? That's how bitter I am

You gotta learn how to register better and be prepared.

Go to ode (1 ms/500mbps up&down) > Schedule finder > "Build" your schedule of the classes you want > Register for this schedule > at ~5:59:55 start clicking it a bunch.

You won't get banned for clicking too many times because the page takes ~1 second to load each time and it should register your schedule even ahead of time. I've done this both quarters so far, and so have my friends and we've all gotten classes that have <4 spots left, and our schedules were submitted seconds before 6 am every time.

I know her ratemyprofessor rating. But she's kind of my only choice right now. What am I getting myself into?

Also, one of the review mentioned that she goes over exactly what's going to be on the test and takes her questions from the book. Is that true?

Thank you all

I had her several years ago. Her class had roughly 30% attendance and virtually nobody liked her. She never did any demos.

On the other hand, she derived every equation and demonstrated more content than was required to do the homeworks. Going to class was never a dull repeat from the textbook.

As 122 is primarily equations (and the left-hand rule), I valued her attention to the math rather than the shiny demos.

Just depends on how you learn. If you like math, you'll love her. If you don't ... You're going to be very unhappy, so I suggest you learn to like math.

I had her for 123 as well. She doesn't use electronic lecture notes, repeatedly derived equations incorrectly in lecture, never held office hours, doesn't understand most student questions, and in general doesn't make it particularly useful to go to lecture. Overall, Goussiou's the worst professor I've had by a long shot. As far as her tests go, she takes her MC questions from the book, so if you do all of the problems there you'll be mostly fine, but they'll still be some what hard.

If you could take 122 another quarter, then I would recommend taking it with a better prof. You can do well in the class (because everyone will be just as disadvantaged having her) but you wont really be learning the material all that well, and the book isn't all that helpful on it's own. I would be careful about getting too far behind if you're considering a physics major though.

People will often have a friend with a higher class standing, who therefore registers earlier, hold a spot for them in a popular class. The friend then drops the class when the first person can register. So really that person getting the spot often knows there is space available before the notification is even sent out to the masses, since they watch their friend drop it. Thus they get into that opening what seems like instantly. This isn't the case every time, but it does happen quite a bit.

f you can't find someone willing to switch with you, learn to be quick on the draw and keep trying. It sometimes just comes down to luck. I remember trying to get into some high demand class my freshman year (I forget which) and I remember I felt discouraged because I always entered in the code within a few seconds after the text came, but never got the class. Then one early Sunday morning over break, I got the notification text and I begrudgingly entered in the code, expecting to not get it. It probably took me a whole minute or two this time to enter in the code. Then, boom, it went through.

Just keep trying, bud.

Something no one else has mentioned is that the emails/texts are sent out in waves, I assume to not break the server; sometimes the text has arrived 5 minutes before the email, and others the exact opposite. Which suggests that others may simply be getting the notifications before you. So, luck really.

Hey, at least we have the notification system now. A few years ago it was just a complete crapshoot.

Curious what admissions decisions have been sent out thus far?

Edit: EE decisions sent
Edit: CSE decisions sent
Edit: ChemE decisions sent
Edit: HCDE decisions sent

I'm a commuter taking about 1 hour one way to get to campus. Due to my poor academic performance and GPA, I've been suggested and advised by many people to seriously consider living closer to campus, ideally in dorms (since obviously they are the closest housing to campus). I want to start living close to campus or dorming from next quarter. I always lived with my family at home, so I have absolutely zero experience with dorms or off-campus housing. I know most people dorm freshman year then go off-campus next year with the friends they made, but most of my friends are dorming or other commuters. But I haven't had that experience. In that situation, I feel like I should dorm for the first year on campus, but winter quarter is almost over so what I read is that there's an extremely small priority for me if I want to dorm from next quarter. Is that right? HFS website is confusing... At this time, Is it better to seek an apartment off-campus? Would it be hard to do this without any other friends? I heard it was highly recommended to not start off your away-from-home living directly off-campus...

I have more questions but I wanted to start with this. What do you think?

ot to discourage you but if you're getting bad grades and have a bad gpa isn't that more of a problem with what you're doing to prepare yourself for your classes? I understand that time is a constraint when you're commuting because I do it too, I live in federal way which is about 40-1hr away and have gotten damn good grades my last year. I dormed my first quarters here at UW and my grades were SHIT, i almost flunked out of math 120 (120 ffs), fucked up CS, and Chem 142 and this was while dorming. I would do nothing but sleep, go out with friends and skip classes. It starts off easy but eventually it got to "Meh its only one class ill just sleep in"... a quarter later.. fuck I'm going to fail.

Honestly I would try improving on your study habits and stuff first because if you're doing bad at uw right now even if you get a dorm and have bad study habits, you're going to do bad anyways and end up spending a lot more money.

Yup. And this is why I'm thinking of living closer. Trust me, this is a decision I made after talking to two school officials (one was my dept. adviser), consulting some trusted people who are alums, talking to friends and mentors. I'm already trying to optimize my time. I've learned that studying alone doesn't work for me, so I've started joining study group aggressively. But, I still tend to lean toward going home early. That means I can't join most study groups and definitely not CLUE. I sometimes try it out, but then coming home around 9-11 puts a strain on my mom who has to pick me up from the P&R. I've tried organizing study sessions with other commuters but it always ended with nothing happening, and facebook study sessions are not effective. At home, I have a lot of responsibilities that I naturally get because, well, I'm at home. I also volunteer a lot, which is going to get cut out soon, as much as I hate to do that. I've had to miss a lecture or even come to lab late more than once this quarter due to unusual traffic too. I believe that some people are simply not good at doing "school" and need to put in more hours to study and invest, in order to get the same score as someone else who might only need half the time. Frankly, I suck at exams. So I needed to look for more ways to optimize. As for finances, I'm definitely not wanting to dorm or live off-campus. I'll probably have to take out loans.

tl;dr As a commuter I keep getting out of school responsibilities and I can't take advantage of all the study groups, CLUE, etc...

I was in a similar situation as you. My grades were complete crap Freshmen year, and my 1.5hr commute one way was not helping. I felt like I was wasting three hours everyday going to school and back (since I cannot study on the bus/car). Sophomore year I lived in an apt on Brooklyn and 41st, super close to campus, and my grades improved.

Living nearby campus definitely helped, since I was close to all of the resources and events on campus, but what helped the most was I took classes that I didn't competently hate. I took a bunch of pre-engineering classes Freshmen year like math and chem, and hated every moment of it. I was completely lost and had no idea what I wanted to do. Sophomore year I attended events hosted on campus, found the Informatics major, decided I wanted to pursue a career in UX design, and took Informatics pre-reqs. I didn't enjoy some classes, like STAT 311, but being close to campus made it easy to join study groups, and I didn't have a bunch of responsibilities at home like you do right now. I did retake CSE 143 though during Spring quarter since I did poorly in it Winter quarter, but managed to 4.0 the second time.

Basically, I'm saying living on/nearby campus can definitely help with events, workshops, office hours, saving commute time, etc. As for getting a dorm for Spring quarter, I don't know much about dorms since I never went through the process, but I think you would have a better shot at getting an apt for spring quarter through the housing Facebook group. I think it is also fine to start off your away-from-home life living off-campus, considering that's what I did. If you have no friends to live with, you can find roommates on the Housing page. I did that for my Sophomore year, and my roommates were fine.

eah, I have been looking into this same thing: you're not alone! At first I was into the idea of a dorm, but it looks really expensive for not much advantage over other options, like living in a shared house near campus. Also what if you get stuck in a shitty location paying over $4000 a quarter for room and board plus dining? Also, that dining hall food. I'm not into it...

Personally, I'm going to be searching for a greek house to call my home for spring quarter, and if that doesn't pan out I'm going to try to live in a shared house off-campus. Dorming doesn't seem worth it for me.

The first thing I noticed about this thread though tbh is that you have your major listed as Bioe, how can you have awful grades and be in Bioe?

So my first quarter I did pretty poorly in Chem 142 (sub 3.0) because I already knew most of the material from high school and I really didn't take it seriously. I'm thinking of moving toward a more biology/chemistry related major and I need to know if I should move on to 152 and try to do very well, or retake 142 before continuing the series.

If you continue getting 15s on the homeworks and pull the same grade on the final, you'll finish with a 70.2%

On the other hand, if you manage to bring everything up a little and get 18's on the rest of the homeworks and get an 85% on the final, you'll pull in a 80% for the class.

If you're trying to get into the CSE major, I'm not sure whether a bad grade or switching to S/NS looks better. In either case, you'd probably have to retake it to have any hope of getting into the major. Also, I think the deadline to switch to S/NS was the 21st.

Amen. By comparison, my EE advisers have all been super blunt with me, and I love that. When looking at other programs within the EE department, such as the accelerated MS, they would advise me along the lines of "you should try, but know that we have never admitted anyone with x, y and z GPA in these courses." Like I said before, my fate with CSE was probably sealed after the end of autumn term 2014, but I was lead to keep trying like I still had some hope. In the end, it delayed my graduation a full year, or roughly $47,000 worth of expenses from tuition and housing as I'm an out of state student. That will probably take me a few years to get over, and even longer to pay off.

Student noob here (so please don't crucify me), but looking to get a laptop for school, and wondering what you guys think about the MacBook versus MacBook Pro.

I like the MacBook because it's super slim and portable (which is very important to me), but I know the Pro is the better machine (but is it overkill for what I'll need?).

What do you guys think? Any advice from Informatics or HCDE students or alums would be great.

Thanks for all the help!

I'm neither Informatics or HCDE, but I know students in both, so I thought I'd give my 2 cents. I'd personally go with the Pro. Both computers should be able to handle most of the heavy work and calculations you'll end up doing. I don't think it's overkill to get the Pro. If anything, it just means your laptop will stay relevant and capable for a couple more years than the standard MacBook.

Programming and processing data is pretty central to both Info and HCDE. What immediately jumps out to me is everything from data collection to viewing that data. Your standard MacBook literally just has one USB-C port for everything. Yes, you can buy adapters and what not, but it can be a burden. I anticipate as you enter your major courses, a dual monitor setup will become increasingly more practical for viewing and processing your data and code. Having a couple extra (standard) USB ports plus an HDMI port would also help you with any type of experiment you may be running where data is digitally collected. If you have any intention of doing high volume data processing locally, having more cores will be a plus, and in that case, the point goes to the Pro.

Long story short, you'll probably be fine with both. If you go with the MacBook, be ready to buy some adapters that you may need to use every now and then or at home. With the MacBook, if you need to do high volume data processing with performance in mind (which will probably be only occasionally at best), you'll probably need to use one of the school computers or do it remotely. Again, both are solid and it just depends what you're willing to do to make it work, whether it be buy more adapters or deal with a heavier laptop.

Edit: I should add, HCDE gives you a lot of room to explore, which is awesome. You need to take into account your interests. If you have any intention of moving into the realm of, for example, electrical or computer engineering with HCDE, then you should go for a PC. You'll find that a lot of engineering software will not run on OSX.

Be sure to check not only the Hub lost and found but Lander and McMahon front desks as well. Although eventually all lost and found gets funneled to the hub (after a week or two I believe), if someone from the dorms found it, they may have dropped it off at the dorm front desk.

The only class I thought was during the summer, so the numbers are atypical. There were about 20 enrolled at the start of the class. 2-3 dropped out, and there were 4 that stayed in the class, but I never saw them the entire quarter, including the final.

Unrelated: It actually really messed with the grades for my class, since there are rules you "have" to follow about the grade distribution. But with a significant percentage of the class missing the final (there were more than the 4 I never saw) it was basically impossible for me to meet the guidelines. I was told it is okay to deviate from the norm in cases like this, but it still caused a really weird distribution and bumped up the grades of some students.

Whether it be your general admission chances applying as an out-of-state 4.0 varsity football science bowl champion or how likely you may be admitted to the CSE department with a 1.5 but only because of that one darn CSE 142 class you thought was really quite silly... let away your queries/pleas/general comments here!

Don't hear it discussed as much as the other 300 level Math courses. I'm planning on taking Math 126, INFO 200, and Q SCI 381 and I'm thinking of replacing Q SCI or INFO with MATH 300.

I am currently in AA, but I really want to do finance, particularly in investment banking. I know many engineers who are taking this path, but I have no idea whether different engineer departments would be the factor or not .

Normally I would just talk to an advisor about my dilemma but today is my last day to drop or S/NS a class. I am a freshman and yesterday I received my midterm back for SOC371 and received a 47.5 out of 60 (79%). The midterm was 20% of my grade. I have nearly 100% in other aspects of the class. Unfortunately the final is 35% of my final grade and will be similar format to the midterm. I studied hard for the midterm and the final may be a similar result. I don't know if I should risk another C+. I emailed my teacher asking about her grading scale but she hasn't emailed back yet so I may have to make the decision without that information. I am applying to Fosters and I'm not sure how using my annual drop would look compared to getting a low grade for the class. Thanks for the help and input!

I'm thinking of taking it next quarter. I'm really interested in web development and would love to learn more about it. However, I've heard that the course is incredibly tough, and if I don't do well, it can hurt my chances of getting into the CSE department. Is this true?

How easy is the course? I'm looking to increase my CGPA. How good is Allison Obourn? Her rate my professor isn't impressive, but I hear good things from people who are currently in her section. Are lectures panopto recorded ? (The classes are at 8:30am for next quarter, so...)

The general consensus on 154 seems to be that it's a basic intro course, and you could learn the material easily enough on your own. While this is true, I personally found 154 to be extremely useful in kick-starting my interest in web development. The languages I learned, especially HTML/CSS/JavaScript, proved to be very useful at the internships I've had, and in classes such as Compilers (401) where we have to implement a JavaScript interpreter.

I took it with Allison 2 years ago, and when I had her the lectures weren't particularly interesting (mostly just an info dump), but the quiz section time was helpful. I'd recommend taking it, especially if you're interested in web dev.

100% would not recommend taking it. As someone who is taking it now, I have to say it is one of the least organized classes I have ever taken.

The class is broken up into 3 components, lecture, quiz section, and a lab portion. It feels like there is 0 communication between these 3. When I took the 14X classes, the homework was assigned either Wednesday or Friday, all the material needed was covered by Friday, and due on the Friday of the next week. With this class , our homework is assigned on Friday, due on the next Wednesday and, sometimes, we don't cover the material needed until Monday, 2 days before its due date. The section has section homework that is due on Tuesday, but is only made available to the students Monday, only a single day to work on it. There have been times where the relevant slides have not been posted, so you either have to check last quarter's website (which is a completely different order from this quarter's, so you end up scrolling through a ton of slides) or you have to just google it. Normally, I wouldn't be opposed to that and would even encourage it, but when you don't have the faintest idea where to start, it's a bit tricky.

They just restructured the course this quarter, so the homework assignments are a different order than they were last quarter and you can tell that it feels kind of funky. I don't think they calculated the difficulty on the assignments too well. If you go to the WPL (web programming lab, the unloved child of the programming lab family) days after an assignment is due, you can routinely find plenty of people working on homework that's been due for a day or two. And not just a small amount of people either. That's another thing, for the first couple weeks, the WPL did not have a set location. In my experience, I will say that if you know exactly what you think is wrong in a your homework, the TAs will be able to assist you. However, I've heard from some friends that some TAs will just tell you to google it, which is honestly pretty ridiculous if it's not an extremely basic question.

On the topic of TAs, I've heard that the grading is incredibly inconsistent compared to the 14x classes. I've had friends who got points taken off for stuff that my TA didn't take points off. The way sections are set up is that you come to a 50 minute class, turn in your homework, and then the TA works on some section projects. At least in my section, there isn't really any group work, a sharp contrast to my 14x sections. You can tell that some people are only there because attendance is mandatory, something I'm not a huge fan of. Sometime what I'll do is I'll work on the section projects by myself while my TA works with the class on the section projects. I hate to sound like an elitist, but section is kind of geared to the lowest common denominator and trivial parts take much longer than they have to. However, it's a nice way to reinforce some concepts, it's just somewhat mind numbing.

As for lectures, I don't think they're set up in the best way. Another comment here mentioned how they feel like it 's an information dump, and I'm inclined to agree. You can tell that Allison is still getting her footing and her timing is still a bit off. She fields incredibly basic questions, which I think the proper response would be to read the slides, google it, or talk to her after class, that she instead tries to give a somewhat in-depth answer and it honestly feels like 1 person slows down the lecture by a lot. This forces her to go through the slides at an accelerated pace to make up for lost time, and you can tell that it's rushed.

With all that being said, learning the topics covered in this class helped me land my current job. The material itself is solid, but the way its presented leaves a lot to be desired. If I had the option of doing it over again, I would not have taken this course and instead learned the topics by myself, possibly with the 154 material.

TL;DR: Learn what the course covers yourself. Don't subject yourself to this very unorganized class.

Hello, are there any informatics majors that feel comfortable commenting or messaging me their grades in their prereqs and other relevant classes as well as their cumulative GPA? Also if possible please share any extracurriculars that you put on your application. I'm applying this spring and am trying to get a good gauge on what it's gonna take to get in. Thanks!

I'm enrolled in PHIL120 and I'm not doing so hot in the class. I'm not gonna fail it or anything, but I'm worried about it affecting my GPA in a negative way (as I'm trying to get into Foster).

Would S/NSing this class hurt my chances of getting into Foster at all? I don't want to have the course be a black mark on my GPA but I also don't want the admissions offers to think poorly of me because I S/NSed a class.

Best: Close to everything you could possibly want in King County.

Worst: Very little math or analysis in the Liberal Arts degree programs...

I'm a sophomore here and I applied upper admission to chemical engineering and got an email yesterday saying I am waitlisted. I didn't even know it was possible for a department to waitlist someone. Does anyone have any experience getting off a department admission waitlist, and, if possible, specifically for chemical engineering? And does anyone know how many people departments usually waitlist? I'm going to go talk to Dave (the chemical engineering advisor) next week but I thought it might be worth a shot posting here to see if anyone has been waitlisted and what it was like, because I've never heard of a department waitlisting someone.

Thanks!

Not sure how many people, sorry.

```
But, you should also investigate other engineering majors.
I'm in MSE now, which covers some different topics than chemical engineering, but it still has aspects of ChemE.
     And honestly, I like it more because (this may not be true) you get to work on the applications of the
     products you make, compared to ChemE where you're a bit more focused on the processes of making stuff.
You may also want to try BSE, which focuses more on chemical engineering of bioresources, etc. Bioengineering may
     also be viable, there's some chemical engineering in there as well.
Haven't taken the course, but if you're planning on doing much computer science in the future, then skip it. If, on
     the other hand, you're just interested in trying out something new then I would absolutely suggest taking it.
     The basic concepts of computer science are relevant to plenty of other fields, plus it's just really fun.
     Also, basic knowledge of a single scripting language is an exceptionally useful skill for anyone, even if you
     never study more advanced concepts in CS.
I've been hearing that participation grades are a myth. It's just a way for TAs/Professors to give you points for
     attendance.
I've personally noticed that my TA will only take notes on who is present in class. After that, during discussion,
     he doesn't mark anything, or write anything down.
Thoughts?
I'm debating whether or not I should take EE 233 or AA 260 (which allows me to apply to AA department). Which one
     is harder/has more work? I'm planning to apply to EE as well so I figured taking as many EE classes would be
     better since EE is my top choice (I am currently taking EE 215)
Confirming that Hermanson is an amazing professor. I won't lie, AA 260 is hard, and Aero is a very, very tough
     major. If you're not dead-set and committed to it (which I question, considering you're not sure if you want
     to take thermo), you'll have a tough time come junior year.
That being said, I'd say take thermo because of a good professor and that it allows you to apply to A&A. 233 on the
     other hand, doesn't provide those benefits.
Not to mention thermo is nice to have under your belt in general.
Hey guys! I am thinking of applying to UW seattle and I need help on the personal statement!it is written that i
     need to include cultural understanding and I find that part of the essay the most difficult one
```

References

[1] Rolf Boldrewood. *In Bad Company and other stories*. The Project Gutenberg EBook. 2016. URL: https://www.gutenberg.org/files/51314/51314-h/51314-h.htm.

[2] Dan Brown. *The DaVinci Code*. text of the actual book. 2003. URL: http://www.amazon.com/The-DaVinci-Code-Dan-Brown-ebook/dp/B007THA4FI.

[3] Charles Dickens. *A Christmas Carol*. The Project Gutenberg EBook. 2004. URL: https://www.gutenberg.org/files/46/46-h/46-h.htm.

[4] Charles Dickens. *American Notes*. The Project Gutenberg EBook. 2013. URL: https://www.gutenberg.org/files/675/675-h/675-h.htm.

[5] Charles Dickens. *The Mystery of Edwin Drood*. The Project Gutenberg EBook. 2010. URL: https://www.gutenberg.org/files/564/564-h/564-h.htm.

[6] Walter Emanuel. *A Dog Day*. The Project Gutenberg EBook. 2016. URL: https://www.gutenberg.org/files/51306/51306-h/51306-h.htm.

[7] Megan Geuss. *France says AZERTY keyboards fail French typists*. French AZERTY keyboard. 2016. URL: http://arstechnica.com/tech-policy/2016/01/france-says-azerty-keyboards-fail-french-typists/.

[8] Jim Harmon. *Break a Leg*. The Project Gutenberg EBook. 2016. URL: https://www.gutenberg.org/files/51320/51320-h/51320-h.htm.

[9] Joseph Heller. *Catch-22*. text of the actual book. 1961. URL: http://www.amazon.com/Catch-22-Joseph-Heller-ebook/dp/B007THA4FI.

[10] Victor Hugo. *Les Miserables*. The Project Gutenberg EBook. 2008. URL: https://www.gutenberg.org/files/135/135-h/135-h.htm.

[11] Stephen King. *The Shining*. text of the actual book. 1977. URL: http://www.amazon.com/The-Shining-Stephen-King-ebook/dp/B007THA4FI.

[12] Marie Lebert. *Entretiens*. The Project Gutenberg EBook. 2008. URL: http://www.gutenberg.org/cache/epub/27035/pg27035-images.html.

[13] Harper Lee. *To Kill a Mockingbird*. text of the actual book. 1960. URL: http://www.amazon.com/To-Kill-A-Mockingbird-Harper-Lee-ebook/dp/B007THA4FI.

[14] Curtis Miller. *Identifying Gender of Authors An application of Markov chains to textual analysis*. Identifying gender. 2015. URL: http://ntguardian-eportfolio.weebly.com/uploads/5/8/5/8/5858892/math_5050.docx.

[15] Alan Nourse. *Primal Difference*. The Project Gutenberg EBook. 2016. URL: https://www.gutenberg.org/files/51321/51321-h/51321-h.htm.

[16] Joseph Toussaint Reinaud. *Invasions des Sarrazins en France*. The Project Gutenberg EBook. 2013. URL: https://www.gutenberg.org/files/43306/43306-h/43306-h.htm.

[17] J.K. Rowling. *The Casual Vacancy*. text of the actual book. 2012. URL: http://www.amazon.com/The-Casual-Vacancy-J-K-Rowling-ebook/dp/B007THA4FI.

[18] J.K. Rowling. *The Cuckoo's Calling*. text of the actual book. 2013. URL: http://www.amazon.com/The-Cuckoo's-Calling-J-K-Rowling-ebook/dp/B007THA4FI.

[19] Anya Sostek. *Duquesne professor helps ID Rowling as author of 'The Cuckoo's Calling'*. Found out who wrote the text. 2013. URL: http://www.post-gazette.com/education/2013/07/16/Duquesne-professor-helps-ID-Rowling-as-author-of-The-Cuckoo-s-Calling/stories/201307160124.

[20] Jimmy Stamp. *Fact of Fiction? The Legend of the QWERTY Keyboard*. What came first: the typist or the keyboard. 2013. URL: http://www.smithsonianmag.com/ist/?next=/arts-culture/fact-of-fiction-the-legend-of-the-qwerty-keyboard-49863249/.

[21] Philipp Von Hilgers. *THE FIVE GREATEST APPLICATIONS OF MARKOV CHAINS*. Markov chain. 2006. URL: http://langvillea.people.cofc.edu/MCapps7.pdf.

[22] David Zax. *How did computers uncover J.K. Rowling's Pseudonym?* Forensic linguistics can use powerful programs to track written text back to its author. 2014. URL: http://www.smithsonianmag.com/science-nature/how-did-computers-uncover-jk-rowlings-pseudonym-180949824/?no-ist.