Ethan James
33378430
Epjames

HW11

Recipe1:

To craft recipe one, I had to look at the from line. I noticed that there were many senders all with the same syntax. The syntax was: <a….@…..>. All I decided to look for was that it started and ended with the open arrows. I then looked for any characters up until the @symbol. I then made sure there was an @symbol. Then I looked for characters after the @symbol that weren't an @symbol that finally ended with an arrow. This indicated the end of a sender address. I made sure the regex expression checked for an email in this format right next to it to make sure that there was more than one From: recipients. This made sure that it matched the format of every email from the recipe 1 emails that were to be sorted.

Recipe2:

In recipe 2, we were given a different set of emails to filter out. I noticed that all of these emails had distinct subject lines that I could use to filter these emails out. I did this by analyzing the subject line in my regex expression. Then I had it search for keywords. I found that each email's subject line contained a distinct keyword no other email had in it's subject line, or it had a misspelled word. I put all these words in my rule to create my recipe. The rule looks for either these keywords or the misspelled words that exists in the subject line.

Recipe3:

I noticed that in the third recipe, it all has to do with some sort of promotion or gambling. We are analyzing the body. There were not that many emails to go through, but I did notice that they all had the words FREE or GEMSTONE. They were trying to show their promotions and convince you to gamble. I made sure my regex expression searched for these words in the body of the email. This helped to filter out these specific emails.

Recipe4:

This one was a little tricky. I did adapt the code from Professor Avi Kak's lecture slides. The first thing I noticed was that all these emails did have the context type of text/html, so this is the first thing I look for. You can see my specific syntax in my code, but it looks to see if the email is of a text/html type. This does not affect the earlier emails since they have already been filtered. Then for the charset part, This part utilizes a negative lookahead (!) to specify that the match should not be followed by either "charset=us-ascii" or "charset=iso-8859-1". Emails containing this are emails I want or emails that are contained earlier in the spam. This just helps to make sure I am not filtering wanted emails into the recipe4 folder. Part four was entirely adapted from AVI KAK lecture slides.