HW 2 CMSC 456. Morally DUE Sep 16
NOTE- THE HW IS SIX PAGES LONG

1. (0 points) READ the syllabus- Content and Policy. READ my NOTES on ciphers and on English. What is your name? What is the day and time of the midterm?

2. (15 points) Klingons use an alphabet of 35 letters. Vulcans use an alphabet of 36 letters. Romulans use an alphabet of 37 letters. Spock notes that Vulcans have an easier time using the Playfair cipher than Klingons or Romulans. He is correct.

   (a) (5 points) Why is it easier for Vulcans to use the Playfair cipher than Klingons or Romulans?

   (b) (5 points) For Klingons to use the Playfair cipher what do they need to do initially?

   (c) (5 points) For Romulans to use the Playfair cipher what do they need to do initially?

GO TO NEXT PAGE

3. (20 points) Alice and Bob are using the Vig Cipher. They RANDOMLY generate a 3-letter word and a 4-letter word and then use that trick in class (Sept 4 lecture) to get a 12-letter word out of it.

   (a) (5 points) The key phrase generated is *bbb cccc*. What is the new 12-letter key phrase?

   (b) (5 points) Encode the word *Nathan* with the key phrase obtained from part (a) using the Vig cipher.

   (c) (5 points) Is there another name for the cipher you are using?

   (d) (5 points) You may have noticed that this is not as strong as Vig is supposed to be. Is there something Alice and Bob could do to avoid this kind of case?

   (e) (0 points) Should Alice and Bob take your advice?

GO TO NEXT PAGE

4. (30 points) This is a programming problem. You will write a program that performs the following tasks, outputting 52 lines in total.

   (a) Input (from standard input) a string of (English) text. For this assignment, you will be processing all letters that appear in the text. The string of text input may contain non-alphabetic characters (e.g., punctuation, whitespace, etc.), so you should discard / ignore any such characters you encounter. You may assume that all the text is contained in a single line of input. We consider lowercase and uppercase letters to be equivalent.

   (b) Recall how we associate each letter with a corresponding number (e.g., $a \mapsto 0, b \mapsto 1, \ldots, z \mapsto 25$). For each $0 \le c \le 25$, compute $B[c]$, which denotes the proportion of times c appeared in the text. For instance, if the string has 40 letters and c appears 17 times, then $B[c] = 17/40 = 0.425$.

       Once you have done this, go through each $0 \le c \le 25$ (starting with $c = 0$ and ending with $c = 25$) and print (to standard output) $B[c]$ on its own line. (So you should print 26 lines for this part.) Make sure you are printing each $B[c]$ as a decimal (i.e., not as a fraction).

   (c) For each $0 \le s \le 25$, compute array C_s, the circular shift of array B from part (b) by shift s (i.e., for each $0 \le k \le 25$, $C_s[k] = B[k + s \pmod{26}]$). Now, go through each $0 \le s \le 25$ (starting with $s = 0$ and ending with $s = 25$) and print the dot product of C_s and B on its own line. (Recall that the dot product of two vectors u, v of length n is defined as $u \cdot v = \sum_{k=1}^{n} u_k v_k$.)

       Note: We expect to find that shifting by 0 results in a dot product of roughly 0.065 and shifting by anything else results in a value of roughly ≤ 0.045. If you do not get this then recheck your work, but it may still be correct if the input text is unusual in some way.

YOU ARE NOT DONE! GOTO NEXT PAGE TO SEE WHAT YOU RUN THIS PROGRAM ON

Run your program on the text from

`https://pastebin.com/raw/wwUeULYU`

What was the largest dot product value you obtained form part (c)? What was the second largest? Report these values along with the rest of your homework. Your code should be uploaded separately (see below).

You are free to choose from various programming languages to complete this problem. By default, we support C, C++, Java, Python2/3, and Ruby. Ask on Piazza if you want more options. You will be submitting all code files you used to complete this problem to the Gradescope assignment called "hw02 - problem 4". Since you will probably want to submit multiple files, you should merge all files into a single zip file and submit that zip file to Gradescope. Upon submission, your code will be automatically run on a Linux machine and tested against various test cases to ensure correctness. You are allowed to submit your code as many times as you want.

Regardless of the language you choose, your submission must include a bash script called `run` (with no file extension). This file must begin with the shebang `#!/usr/bin/env bash` on the very first line. This script will be run each time the autograder tries to run your code, so add to this file any commands that are needed to run your code. This gives you greater flexibility regarding how you want to organize your code. Additionally, if you are using a non-scripting language such as Java, also upload a bash script called `build`, also with shebang. This script will be called once upon submission to compile your code before execution.

If you have any questions or confusions, or if you encounter any technical difficulties, feel free to ask for help on Piazza.

GOTO NEXT PAGE

5. (20 points) Let

$$M = 2^2 \times 3^3 \times 11 \times 197$$

(a) (5 points) How many positive factors does M have? (Include 1 and M itself.)

(b) (15 points) Find all positive factors of M that are between 7000 and 9999.

Do it by hand and show your work. It is NOT enough to show one such number, you must show all such numbers and prove there are no other ones. Also, DO NOT check all factors using brute force; you should use reasoning to reduce your search space.

(NOTE- these numbers are different from the ones on the slides.)

GO TO NEXT PAGE

6. (15 points) (Please do by hand – the numbers do not get that big. I cannot stop you from using a computer; however, you will get more out of the exercise if you do it by hand.)

   (a) (5 points) Which numbers in $\{1, 2, 3, \ldots, 17\}$ have an inverse mod 18?

   (b) (5 points) For all such numbers, give the inverse.

   (c) (5 points) Show that, for all n, the inverse of $n - 1 \pmod{n}$ is actually $n - 1$.