CSC 333: Cryptology: 2020 Spring
Assignment #1

Last Modified 2020 April 3

Purpose:

To go over:

- Basic cryptology
- Basic cryptanalysis

Computing

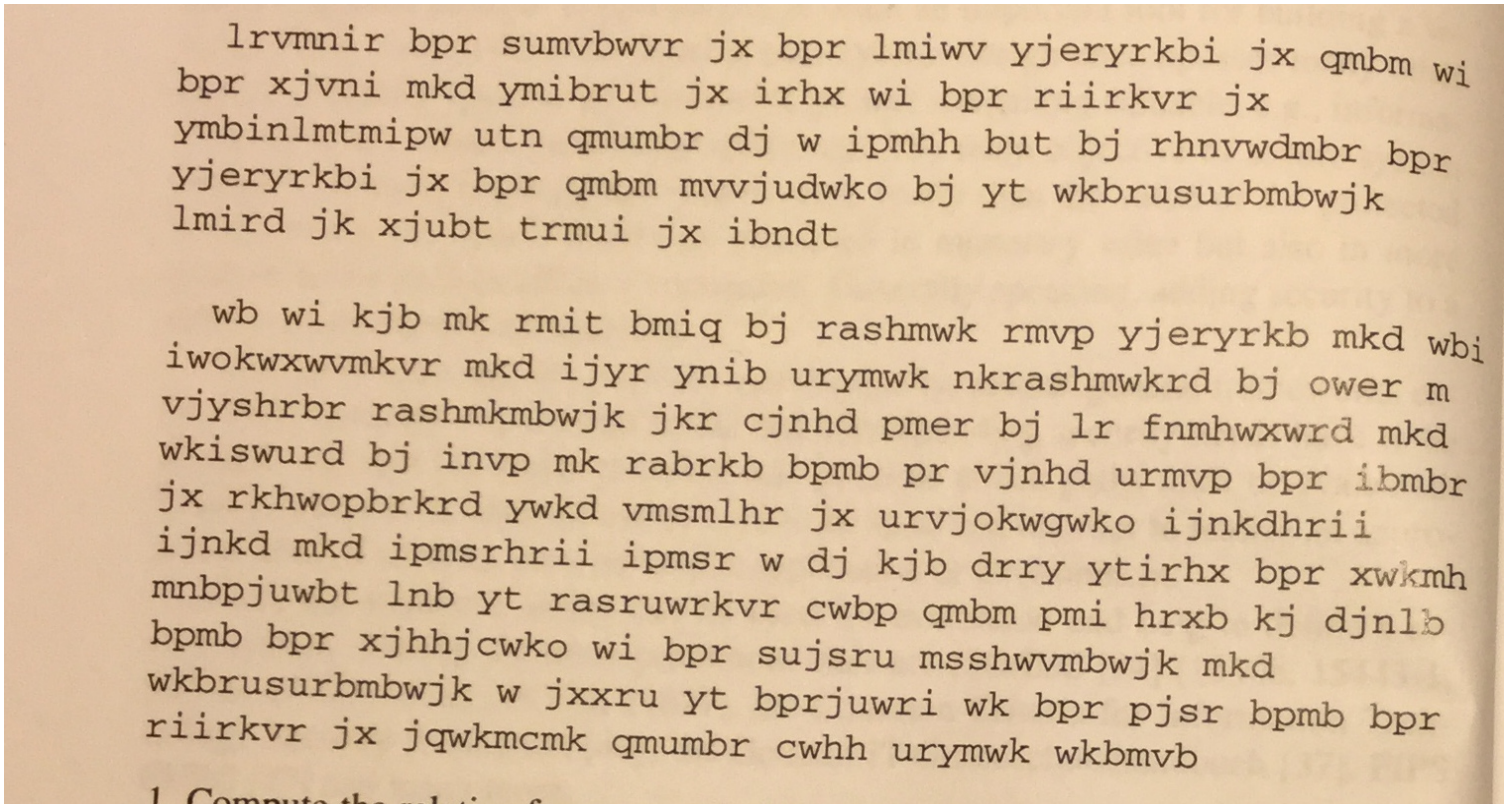
Please limit yourself to one of the following languages:

- Python
- Java
- C/C++

I. **Assignment:**

1. (From Paar and Pelzl)

Please decrypt the text



lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx qmbm wi bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jx ymbinlmtmipw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr yjeryrkbi jx bpr qmbm mvvjudwko bj yt wkbrusurbmbwjk lmird jk xjubt trmui jx ibndt

wb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi iwokwxwvmkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m vjyshrbr rashmkmbwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd wkiswurd bj invp mk rabrkb bpmb pr vjnhd urmvp bpr ibmbr jx rkhwopbrkrd ywkd vmsmlhr jx urvjokwgwko ijnkdhrii ijnkd mkd ipmsrhrii ipmsr w dj kjb drry ytirhx bpr xwkmh mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj djnlb bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd wkbrusurbmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr riirkvr jx jqwkmcmk qmumbr cwhh urymwk wkbmvb

1. Compute the

- Compute the relative frequency of all letters `A..Z` in the ciphertext. You may want to use a tool such as the open-source program `CrypTool` for this task. However, a paper and pencil approach is still doable.
- Decrypt the ciphertext with the help of the relative letter frequency (and word frequency, doubled letter frequency, etc) of English. Note that the text is relatively short and that the letter frequencies in it might not perfectly align with that of general English fromteh table.
- Who wrote it?

2. (From Paar and Pelzl)

Compute the following without a calculator:

a. 15*29 mod 13
b. 2*29 mod 13
c. 2*3 mod 13
d. -11*3 mod 13
e. 1/5 mod 13
f. 1/5 mod 7
g. 3*2/5 mod 7

3. **Programming:**

Prof Joe gave the following program in class to encrypt and decrypt with the affine cipher:

```
# affine.py

from random import *

def stringToIntList(text):
    text = text.lower()

    lst = [(ord(letter) - ord('a')) for letter in text if letter.isalpha()]
    return lst

def intListToString(lst):
    text = ''.join([chr(ord('a')+code) for code in lst])
    return text

#affine cipher

def affine_h(pt,a,b):
    #helper function: works on coded text
    return [(a*x+b)%26 for x in pt]

def antiAffine_h(pt,inverseA,minusB):
    #helper function: works on coded text
    return [((x+minusB)*inverseA)%26 for x in pt] # Change that first x!

def encode(pt,a,b):
```

```
    pt = stringToIntList(pt)
    return intListToString(affine_h(pt,a,b))

def decode(pt,inverseA,minusB):
    pt = stringToIntList(pt)
    return intListToString(antiAffine_h(pt,inverseA,minusB))
```

Prof Joe encrypted a string with an unknown `a` and `b` to get `'gjccz'` Write a Python program to try every possible a^{-1} and `-b` to figure out what Prof Joe said.

Please submit:

- Your program
- What `'gjccz'` means
- What you found as a^{-1} and `-b`