رامتین احسانی – 97521018

پروژه اول درس امنیت سیستم های کامپیوتری

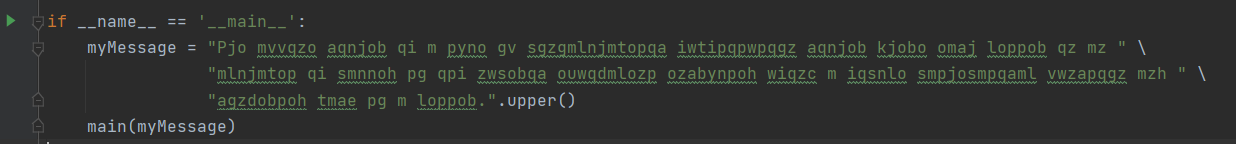
شکستن رمز های Affine:

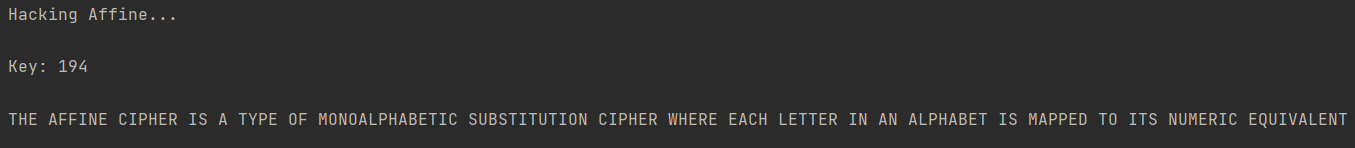
برای شکستن این رمز ها از روش brute-force استفاده کردم. در Affine برای رمز گذاری داشتیم:

(ax + b) modulo 26

پس با تست کردن عدد های مختلف a و b با brute-force در نهایت این رمز شکسته خواهد شد.

در کدی که نوشتم این ciphertext به این صورت شکسته خواهد شد:





بعد از تست هر کلید، باید راهی برای تشخیص انگلیسی بودن یا نبودن متن داشته باشیم. برای اینکار فایل EnglishDetect.py را ساختم. برای تشخیص انگلیسی بودن، متن را به تابع isEnglish میدهیم. این تابع متن را واژه به واژه جدا میکند و character های non-letter را با regular expressions حذف میکند. علت حذف این character ها بخاطر این است که اگر حذف نشوند باعث گمراهی الگوریتم میشوند. برای مثال، “Great!” کلمه ای انگلیسی است ولی علامت “!” باعث میشود که در بانک کلمات ما پیدا نشود.

بعد از حذف character های اضافی، در بانک کلماتی words.txt جستجو میکند و با هر کلمه ای که در بانک پیدا میکند، تعداد match ها را افزایش میدهد.

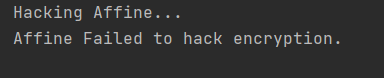
در نهایت برای هر متن درصدی را باز میگرداند و در صورتی که این درصد از مقدار از پیش تعیین شده ای مانند 20 درصد بیشتر باشد، آن را به عنوان متن انگلیسی قبول میکند.

برای تشخیص انگلیسی بودن، برای محکم کاری، از پکیجی در پایتون به نام textbolb هم استفاده کردم که در ادامه آن را توضیح میدهم.

پس در affine، با تست هر کلید و تشخیص انگلیسی بودن یا نبودن، اگر الگوریتم رمز گذاری affine بوده باشد، در نهایت به جواب میرسیم.

الفبای تعریف شده هم برابر با ABCDEFGHIJKLMNOPQRSTUVWXYZ است و ورودی هرچه باشد، با استفاده از upper() به upper-case تبدیل میشود و خروجی هم به صورت upper-case است. اینکار برای سادگی الگوریتم انجام شد.

در صورت پیدا نکردن جوابی با affine، با پیغام زیر روبرو میشویم:



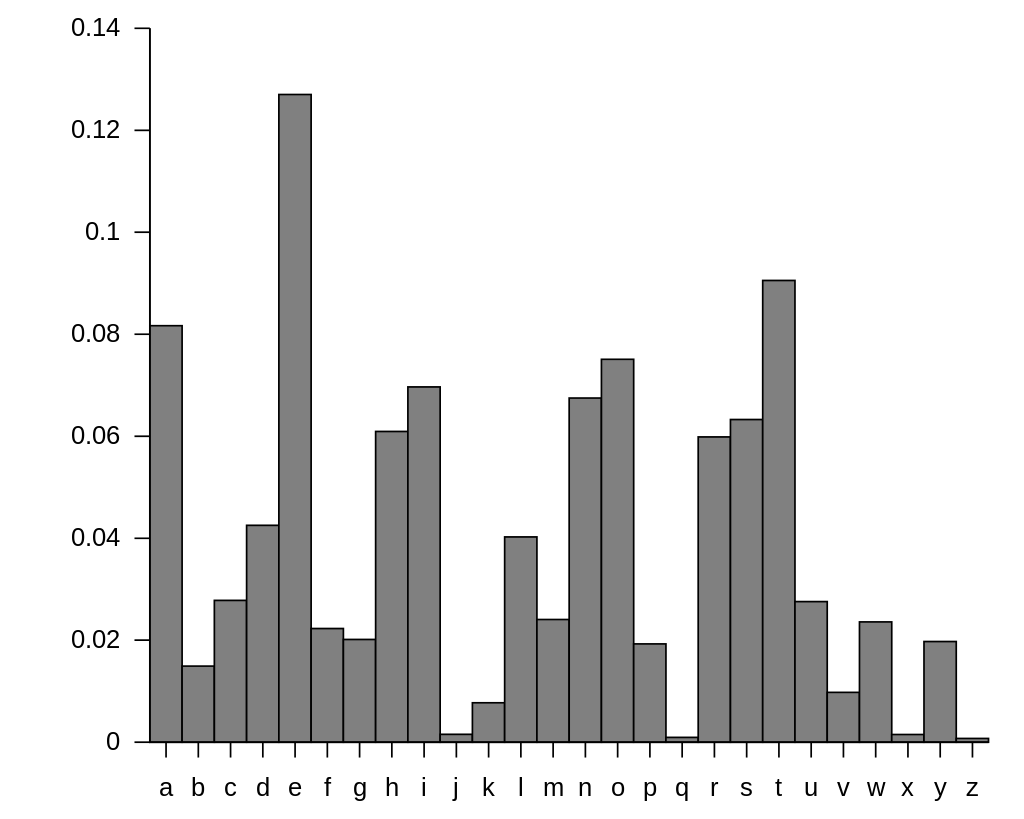
در مرحله ی آخر، صرفا جهت محکم کاری، از textblob برای تشخیص انگلیسی بودن یا نبودن استفاده کردم که در صورتی که این مرحله با شکست مواجه شود، پیغام Possible Warning: blob not detected را میدهد. برای استفاده از این پکیج **باید متصل به اینترنت** باشید.

شکستن رمز های Vigenere:

برای شکستن این رمز ها کار ما سخت تر است چون کلید میتواند حالت های بسیار زیادی داشته باشد.

برای Vigenere از روش Kasiski examination برای پیدا کردن حالت های محتمل اندازه کلید انتخاب شده و از روش تحلیل فرکانسی برای تحلیل متن های بدست آمده با کلید های محتمل برای بدست آوردن کلید اصلی استفاده کردم.

فایل freqAnalysis.py برای تحلیل فرکانسی است. این تحلیل طبق این نمودار زیر انجام شده است.



روش Kasiski examination اینگونه عمل میکند که رشته از حرف های مشابه که در متن تکرار میشوند را پیدا میکند. فاصله بین این رشته ها به احتمال زیاد باید ضریبی از طول این رشته ها باشد. برای مثال:

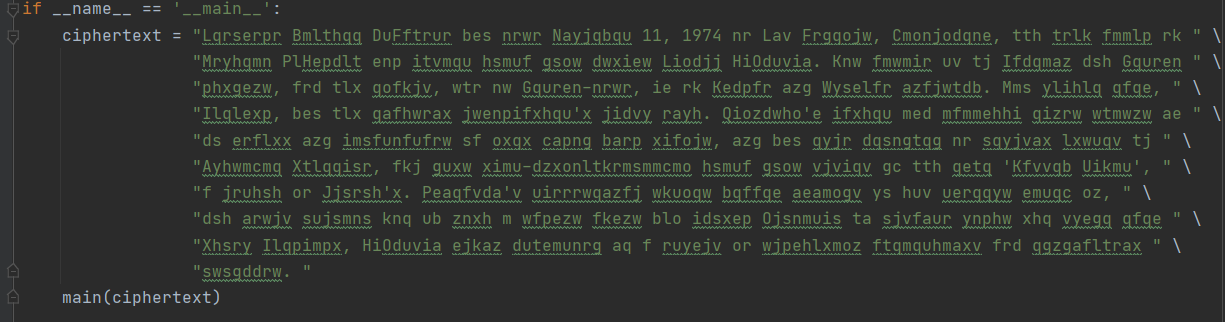


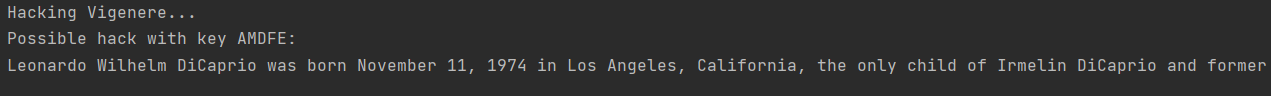
با رمزی مثل abcde با طول 5 کلمه crypto با abcdea همسان میشود.

پس از پیدا کردن طول رمز های ممکن با این روش، کلید های مختلف را امتحان میکنیم و با تحلیل فرکانسی که انجام میدهیم، در نهایت برای هر رمز یک امتیازی بدست میاید.

بر اساس این امتیاز ها کلید های محتمل خود را میسازیم و متن را decrypt میکنیم.

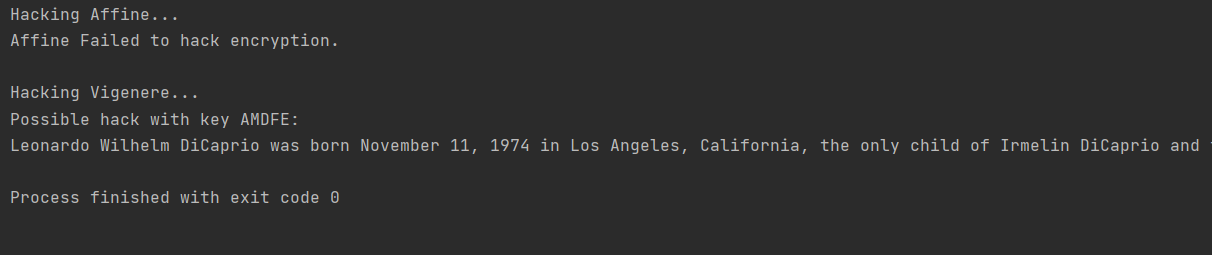
بعد از بدست آمدن plaintext باید تشخیص دهیم که متن انگلیسی است یا نه که روش استفاده شده در قسمت قبلی توضیح داده شد.





رمزگشایی:

بعد از پیاده سازی هر دو روش، هر دو را در فایلی به نام main.py فرا خوانده ام. برنامه را اینگونه نوشتم که اول متن رمز شده را با الگویتم Affine سعی میکند بشکند. در صورتی که موفق نبود، سراغ Vigenere می رود. برای تست برنامه، فایل **main.py** را اجرا کنید و برای تغییر متن ورودی، متغیر ciphertext را در این فایل تغییر دهید. با دادن متن رمز گذاری شده به هر دو تابع با خروجی مانند زیر مواجه میشوید:



در این تست، متن داده شده با کلید AMDFE با Vigenere رمز گذاری شده است. با Affine شکسته نشد ولی با Vigenere با موفقیت شکسته شده است.

همه ی کد ها، بانک کلمات و فایل های نوشته شده برای پروژه در فولدر Codes قرار دارند.