REPORT 9

所属　電子情報工学部　通信
学籍番号　08D18151
氏名　山下慎太郎

- 課題１１－９

  <考え方>

  c_iはj = a × i + b(mod N)で算出されるc_jで置き換えられるため、c_iとc_jの対が２対あれば、a,b が求められる。文字の偏りを用いて二対を推測した。その時に用いたものが、英単語で使用頻度が高いとされている、e,a である。

  <結果>

  課題１１－１０でまとめて記載する。

  <ソースコード>

  課題１１－１０で記載する

- 課題１１－１０

  <考え方>

  プログラムが始まるときに、start に現在時刻を代入し、終わったときに end にその時の時刻を保存し、引き算することで、かかった時間を導出した。

  <結果>

```
PS C:\Users\shintaro\Google ドライブ\9> python 11_10.py
8 or 9 : 8
ratio :  0.9351874244256349 a :  71 b :  56
time :  48.208975076675415
PS C:\Users\shintaro\Google ドライブ\9> python 11_10.py
8 or 9 : 9
ratio :  0.9351874244256349 a :  71 b :  56
time :  0.2625997066497803
```

  ９で作成した関数で実行すると８に比べかなり早くできることが分かった

  単純に考えて、８では for 文が 6500 回近く回っていたが、９では 25 回程度だったので、時間がこれほどまでに変化した原因はここだと考えられる。

  <ソースコード>

  １１－９で作成した関数

```
def estimate(top_de, top_en, second_de, second_en):
    for a in range(N):
        for b in range(N):
```

```
                if top_en == (a*top_de+b) % N and second_en ==
(a*second_de+b) % N:
                    return a, b
    return 0, 0
```

main 関数の一部

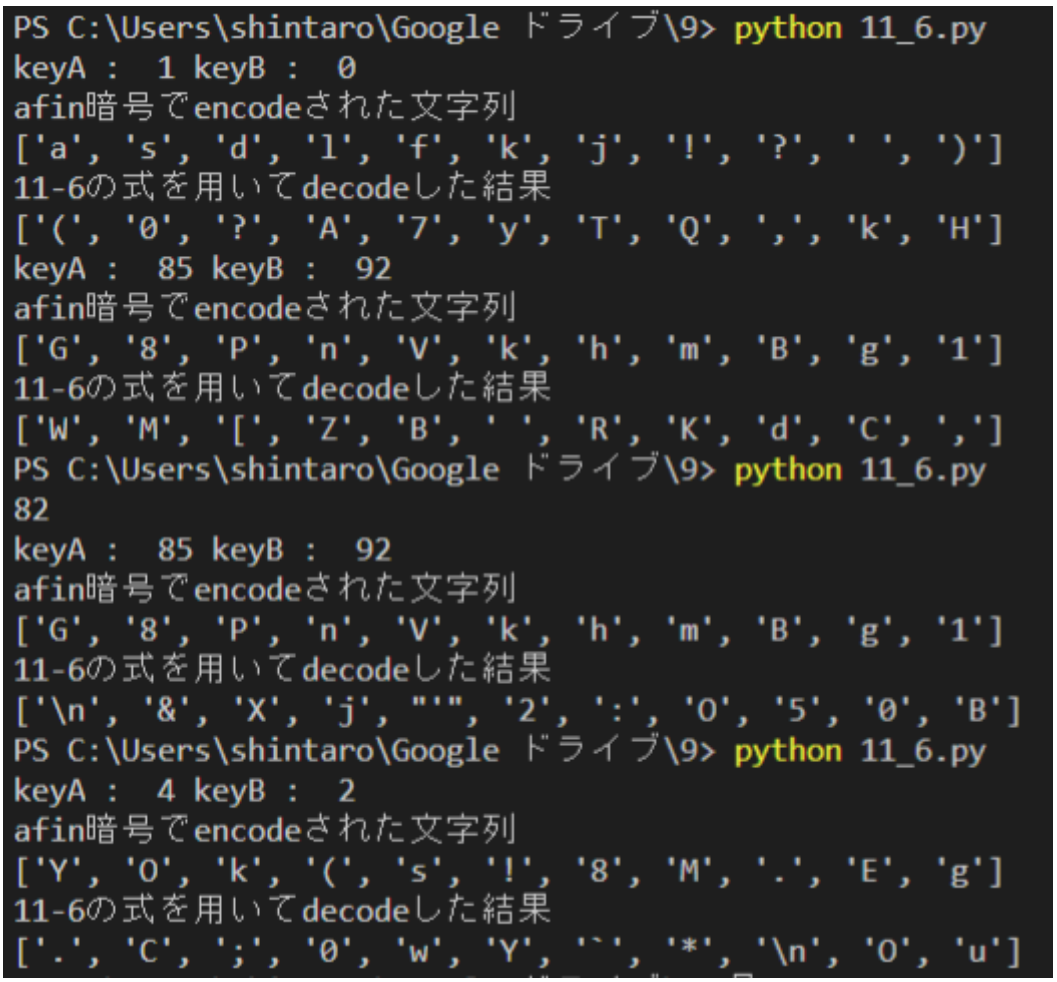
```
        store_data = collections.Counter(chk_txt)
        challacter, num = zip(*store_data.most_common())

        store_rate = 0

        for i in range(5):
            for j in range(5):
                store_1 = STR_LIST.find('e')
                store_2 = STR_LIST.find(challacter[i])
                store_3 = STR_LIST.find('a')
                store_4 = STR_LIST.find(challacter[j])

                a, b = estimate(store_1, store_2, store_3, store_4)
                decoded = decode(chk_txt, a, b)
                rate = chk_ratio(decoded, wordlist)

                if rate > store_rate:
                    store_rate = rate
                    store_decoded = decoded
                    store_a = a
                    store_b = b

        end_time = time.time()
        print('ratio : ', store_rate, 'a : ', store_a, 'b : ', store_b)
    print("time : ", end_time-start_time)
```

- 課題１１－１１

  <考え方>

資料通りの a,b の組み合わせでエンコード、デコードを行いどのような結果が出るのかを調べた。

<結果>
"asdlfkj!?)"が元の平文



(1,0)の組み合わせの時
単純に j=i となり、エンコードしても何も変化しないのは明らか。

a>=N,b>=N の時
modN なので、a の値が N で割った余りと同値になるためおかしな結果が表示される。

Nと互いに素でない a の時
この場合も modNなので、a の値が N で割った余りと同値になるためおかしな結果が表示される。

＜ソースコード＞

```
        keyA = 1
        keyB = 0

        print("keyA : ", keyA, "keyB : ", keyB)

        for i in range(int(len(str_list)/2)):
            afin.append(str_list[(keyA*i+keyB) % N])
            j.append((keyA*i+keyB) % N)   # 番号を記憶

        encoded, encode_j = encode(text, afin, str_list, j)
        print("afin 暗号で encode された文字列")
        print(encoded)

        for i in range(len(encoded)):   # 11-6 の方法で decode
            store_decoded.append(str_list[(x*(encode_j[i]-keyB)) % N])

        print("11-6 の式を用いて decode した結果")
    print(store_decoded)
```