1. The cryptanalysis strategies related to finding the key length

So basically, first we write the program (StatisticsAnalyzer.py) to retrieve the ciphertext from CipherText.txt under the same directory, and then generate subsequences of one every N characters. The regarding code is as shown below:

```
subsequences = {}
cipherlen = len(ciphertext)

#initialize subsequences
for group_index in range(N):
    subsequences.update({group_index:""})

text_index = 0
temp1 = 'a'

#retrieving subsequences
while text_index < cipherlen:
    for group_index in range(N):
        temp1 = ciphertext[text_index]
        subsequences[group_index] = subsequences[group_index] + temp1
        text_index += 1
        if text_index >= cipherlen:
            break

#print subsequences
print("Subsequences are:")
for group_index in range(N):
    print("Subsequence", group_index, ":",subsequences[group_index])
```

And then we calculate the frequencies of each letter in each subsequence and output them to the console as well as Statistics.txt under the same directory. The regarding code is as shown below:

```
#print subsequences
print("Subsequences are:")
for group_index in range(N):
    print("Subsequence", group_index, ":",subsequences[group_index])

temp2 = 'a'
key_guess = ""
flag = True
#analyze subsequences
letter_freqs = []
f1 = open("Statistics.txt", "w")
for group_index in range(N):
    letter_freqs = []
    for letter_index in range (26):
        temp2 = chr(97 + letter_index)
        letter_freqs.append(letter_freq_pair(temp2, subsequences[group_index].count(temp2)/len(subsequences[group_index])))
    letter_freqs.sort(key=lambda x: x.freq, reverse=True)
    f1.write("//////////////////////////////////////\nSubsequence " + str(group_index) + " statistics:\n")
    print("Subsequence", str(group_index), "statistics:")
    for letter_index in range (26):
        f1.write("Letter: " + str(letter_freqs[letter_index].letter) + "   freq: " + str(letter_freqs[letter_index].freq) + " \n")
        print("Letter:", letter_freqs[letter_index].letter, "freq:", letter_freqs[letter_index].freq)
    if ord(letter_freqs[0].letter) >= 101:
        key_guess += chr(97 + ord(letter_freqs[0].letter) - 101)
    else:
        key_guess += chr(ord(letter_freqs[0].letter) - 97 + 119)
    if letter_freqs[0].freq < 0.1:
        flag = False
```

Notice that the code relevant to ‘flag’ was added later on. Also notice that the letter-frequency pairs in the **letter_freqs** list are sorted based on their frequency value (from high to low). Initially, we may run the program with some random N values like 5, and we’ll get stuff like this:

//////////////////////////////////////
Subsequence 0 statistics:
Letter: f freq: 0.05956614509246088
Letter: q freq: 0.051564722617354196
…
//////////////////////////////////////
Subsequence 1 statistics:
Letter: f freq: 0.06241109530583215
Letter: s freq: 0.05369843527738265
…
//////////////////////////////////////
Subsequence 2 statistics:
Letter: f freq: 0.06294452347083926
Letter: u freq: 0.05103129445234708
…
//////////////////////////////////////
Subsequence 3 statistics:

Letter: f freq: 0.059388335704125175
Letter: s freq: 0.05316500711237553
…
//////////////////////////////////////
Subsequence 4 statistics:
Letter: f freq: 0.05815401031477859
Letter: s freq: 0.049617641828205584
…

Notice that none of the highest frequencies in each subsequence is significantly large – in English the letter with highest frequency is ‘e’, with a frequency of ~0.12. So N=5 is likely an incorrect guess. Also, we can start to verify the correctness of the guessed value of N by checking whether the highest frequencies in each subsequence are all, let’s say, greater than or equal to 0.10. In the Python program, this is done by the code relevant to the ‘flag’ variable (as mentioned earlier) as well as a few more lines of code as below:

```
f1.write("//////////////////////////////////////\nKey guessed based on the highest-frequency letter in each subsequence is: " + key_gues
print("Key guessed based on the highest-frequency letter in each subsequence is:", key_guess)
f1.close()
if flag:
    print("Bingo!")
else:
    print("Bruhhhhhhhhh")
print("Statistics analyzation completed. The statistics is stored in Statistics.txt")
```

(Notice that this piece of code also prints the guessed key, which will be discussed later in part 2.) This way, if the highest frequencies in each subsequence are all greater than or equal to 0.10, then the program will print “Bingo!” indicating our guess of N (i.e. key length) is most likely correct. Otherwise, it will print "Bruhhhhhhhhh" indicating a wrong guess.

2. The cryptanalysis strategies related to finding each of the letters of the key

Before we really start to run the program, we may talk about how we guess the key once we get the (mostly) correct key length: notice that the letter with the highest frequency in each subsequence is most likely transferred from ‘e’ in plaintext, right? So we can guess each letter in the key by finding which letter is required to transfer ‘e’ to the letter with the highest frequency in the regarding subsequence. For instance, if the letter with the highest frequency in the first subsequence is ‘f’, then we may guess the first letter in the key to be ‘b’. (Since an

'e' can be transferred to an 'f' by a 'b' in the respective place in the key.) The code in charge of doing so is as shown below:

```
if ord(letter_freqs[0].letter) >= 101:
    key_guess += chr(97 + ord(letter_freqs[0].letter) - 101)
else:
    key_guess += chr(ord(letter_freqs[0].letter) - 97 + 119)
```

OK, now we are done with the 'theoretical explanation' part. Let's try some values of N starting from 1:

N=1:

```
Letter: e freq: 0.039759593157651 41
Letter: v freq: 0.03598990006756997
Letter: p freq: 0.03595433692521071
Letter: b freq: 0.03467406380027739
Letter: h freq: 0.034531811230840356
Letter: d freq: 0.03414061666488851
Letter: m freq: 0.0337138589565774
Letter: k freq: 0.03136669156086632
Letter: i freq: 0.030157544720651516
Letter: r freq: 0.028983961022795976
Letter: z freq: 0.028983961022795976
Letter: l freq: 0.02500088907855898
Letter: x freq: 0.024680820797325653
Letter: y freq: 0.016679113766492407
Key guessed based on the highest-frequency letter in each subsequence is: b
Bruhhhhhhhhh
Statistics analyzation completed. The statistics is stored in Statistics.txt
```

Nope.

N=2:

```
Letter: v freq: 0.03656021054129028
Letter: p freq: 0.03592005121274628
Letter: m freq: 0.03570666476989828
Letter: b freq: 0.03563553595561562
Letter: h freq: 0.03456860374137563
Letter: d freq: 0.03286151219859165
Letter: k freq: 0.03271925457002632
Letter: i freq: 0.030514261327263675
Letter: r freq: 0.029589586741589017
Letter: z freq: 0.027597979941674372
Letter: l freq: 0.024966213813215735
Letter: x freq: 0.02468169855608507
Letter: y freq: 0.017284301870687815
Key guessed based on the highest-frequency letter in each subsequence is: bb
Bruhhhhhhhhh
Statistics analyzation completed. The statistics is stored in Statistics.txt
```

Nope.

…

…

N=6:

```
Letter: b freq: 0.036064874093043105
Letter: v freq: 0.036064874093043105
Letter: m freq: 0.03585147247119078
Letter: p freq: 0.03478446436192915
Letter: d freq: 0.031156636790439608
Letter: k freq: 0.030089628681177975
Letter: z freq: 0.02923602219376867
Letter: i freq: 0.027955612462654718
Letter: r freq: 0.02731540759709774
Letter: l freq: 0.025181391378574478
Letter: x freq: 0.023687580025608196
Letter: y freq: 0.020486555697823303
Key guessed based on the highest-frequency letter in each subsequence is: bbbbbb
Bruhhhhhhhhh
Statistics analyzation completed. The statistics is stored in Statistics.txt
```

Still nope…

N=7:

```
Letter: k freq: 0.009459795867562858
Letter: v freq: 0.007966143888473986
Letter: x freq: 0.0012447099825740602
Letter: q freq: 0.0009957679860592482
Letter: z freq: 0.0009957679860592482
Letter: j freq: 0.0007468259895444362
Key guessed based on the highest-frequency letter in each subsequence is: sobjmca
Bingo!
Statistics analyzation completed. The statistics is stored in Statistics.txt
```

Br…Bingo! Seems that we've found the correct key length N, which is **7**! Also the key is guessed to be **sobjmca**, which makes sense. Now let's further check it by decrypting the ciphertext with it using EncrypterDecrypter.py:

```
PS C:\[redacted]\ECE 458\A1> python .\EncrypterDecrypter.py
Please specify operation. Input 1 for encryption, input 0 for decryption: 0
Please input key: sobjmca
CipherText successfully retrieved
Decryption completed. The decrypted plaintext is stored in PlainText_Result.txt
```

And the result plaintext looks like this:


PlainText_Result.txt

```
1  behindwinstonsbackthevoicefromthetelescreenwasstillbabblingawayaboutpigironandtheoverfulfilmentoftheninththreeyearplanthetelescreenreceiv
   edandtransmittedsimultaneouslyanysoundthatwinstonmadeabovethelevelofaverylowwhisperwouldbepickedupbyitmoreoversolongasheremainedwithinthe
   fieldofvisionwhichthemetalplaquecommandedhecouldbeseenaswellasheardtherewasofcoursenowayofknowingwhetheryouwerebeingwatchedatanygivenmome
   nthowoftenoronwhatsystemthethoughtpolicepluggedinonanyindividualwirewasguessworkitwasevenconceivablethattheywatchedeverybodyallthetimebut
   atanyratetheycouldpluginyourwirewheneverthewantedtoyouhadtolivedidlivefromhabitthatbecameinstinctintheassumptionthateverysoundyoumadewas
   overheardandexceptindarknesseverymovementscrutinizedalreadyhesatasstillasamouseinthefutilehopethatwhoeveritwasmightgoawayafterasingleatte
   mptbutnotheknockingwasrepeatedtheworstthingofallwouldbetodelayhisheartwasthumpinglikeadrumbuthisfacefromlonghabitwasprobablyexpressionles
   shegotupandmovedheavilytowardsthedoorforwhomitsuddenlyoccurredtohimtowonderwashewritingthisdiaryforthefuturefortheunbornhismindhoveredfor
   amomentroundthedoubtfuldateonthepageandthenfetchedupwithabumpagainstthenewspeakworddoublethinkforthefirsttimethemagnitudeofwhathehadunder
   takencamehometohimhowcouldyoucommunicatewiththefutureitwasofitsnatureimpossibleeitherthefuturewouldresemblethepresentinwhichcaseitwouldno
   tlistentohimoritwouldbedifferentfromitandhispredicamentwouldbemeaninglesshesatbackasenseofcompletehelplessnesshaddescendeduponhimtobeginw
   ithhedidnotknowwithanycertaintythatthiswasitmustberoundaboutthatdatesincehewasfairlysurethathisagewasthirtynineandhebelievedthathehadbeen
   borninorbutitwasneverpossiblenowadaystopindownanydatewithinayearortwotheministryoftruthminitrueinnewspeakwasstartlinglydifferentfromanyot
   herobjectinsightitwasanenormouspyramidalstructureofglitteringwhiteconcretesoaringupterraceafterterracemetresintotheairfromwherewinstonsto
   oditwasjustpossibletoreadpickedoutonitswhitefaceinelegantletteringthethreeslogansofthepartyatthismomenttheentiregroupofpeoplebrokeintoade
   epslowrhythmicalchantofbbbboverandoveragainveryslowlywithalongpausebetweenthefirstbandthesecondaheavymurmuroussoundsomehowcuriouslysavage
   inthebackgroundofwhichoneseemedtohearthestampofnakedfeetandthethrobbingoftomtomsforperhapsasmuchasthirtysecondstheykeptitupitwasarefraint
   hatwasoftenheardinmomentsofoverwhelmingemotionpartlyitwasasortofhymntothewisdomandmajestyofbigbrotherbutstillmoreitwasanactofselfhypnosis
   adeliberatedrowningofconsciousnessbymeansofrhythmicnoisewinstonsentrailsseemedtogrowcoldinthetwominuteshatehecouldnothelpsharinginthegene
   raldeliriumbutthissubhumanchantingofbbbbalwaysfilledhimwithhorrorofcoursehechantedwiththerestitwasimpossibletodootherwisetodissembleyourf
   eelingstocontrolyourfacetodowhateveryoneelsewasdoingwasaninstinctivereactionbuttherewasaspaceofacoupleofsecondsduringwhichtheexpressionof
   hiseyesmightconceivablyhavebetrayedhimanditwasexactlyatthismomentthatthesignificantthinghappenedifindeeditdidhappenforsometimehesatgazing
   stupidlyatthepaperthetelescreenhadchangedovertostridentmilitarymusicitwascuriousthatheseemednotmerelytohavelostthepowerofexpressinghimsel
   fbuteventohaveforgottenwhatitwasthathehadoriginallyintendedtosayforweekspasthehadbeenmakingreadyforthismomentandithadnevercrossedhismindt
   hatanythingwouldbeneededexceptcouragetheactualwritingwouldbeeasyallhehadtodowastotransfertopapertheinterminablerestlessmonologuethathadbe
```

And this looks good! So we've found the correct key, which is **sobjmca**! Done! :D