İSTİNYE ÜNİVERSİTESİ İSTANBUL

Ceaser Cipher

Vigenere Cipher

Brute Force Attack

Proje Sonuç Raporu

Mayıs 2021

İSTİNYE ÜNİVERSİTESİ İSTANBUL

Yazarlar

Tolgahan Şişman
Raporlar, Kaynak araştırması, Vigenere Cipher Encrypt Kodu, Vigenere Cipher Decrypt Kodu, Brute Force Attack Kodu, Sezar Şifreleme Algoritması, Proje geliştirme

Ali Murat Tava
Kaynak araştırması, Vigenere Cipher Encrypt Kodu, Vigenere Cipher Decrypt Kodu, Brute Force Attack Kodu, Sezar Şifreleme Algoritması, Proje geliştirme

Kaan İnce
Kaynak araştırması, Vigenere Cipher Encrypt Kodu, Vigenere Cipher Decrypt Kodu, Brute Force Attack Kodu, Sezar Şifreleme Algoritması, Proje geliştirme

Danışman
Ahmad Hasan Abed Al Khas

Proje Fikri

Vigenere Cipher:

Vigenere şifresi çoklu alfabeli (Polyalphabetic) özellikte bir blok şifreleme yöntemidir. İlkel bir yöntem olmasına rağmen çoklu alfabeli oluşu ve bu sayede şifrenin harf bloklarına uygulanması onu güçlü kılmıştır. Vigenere şifresi, verileri şifrelemek için 'anahtar(key)' kavramını kullanmıştır. Bu anahtar, şifreleme dünyasına yeni bir boyut getirmiştir, çünkü şifrenin çözülmesinde(decrypt) frekans analizine (harflerin tekrar etme sıklığına göre yapılan şifre çözme yöntemi) direnmeyi başarmıştır. Ancak yine de şifreli mesajın anahtar boyu kadar bloklarına frekans analizi yapılarak mesaja ulaşılması mümkündür.

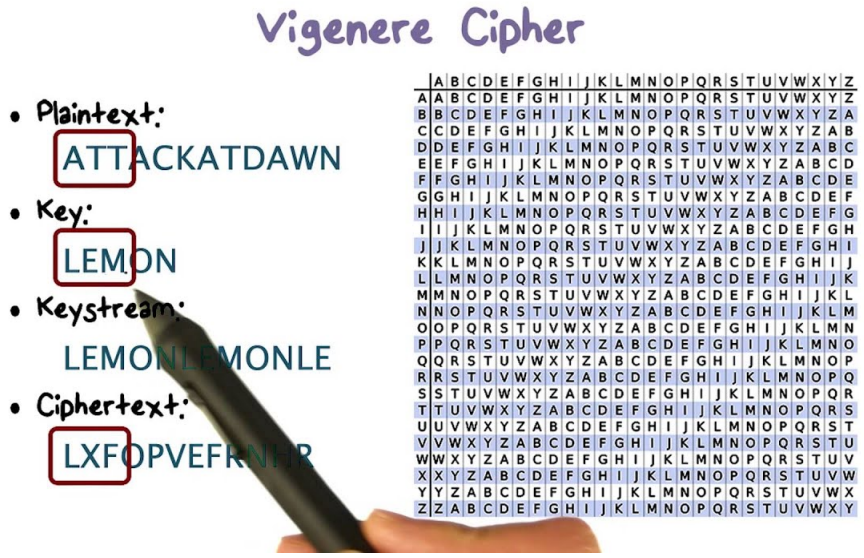


Figure 1: Vigenere Cipher

Orijinal metnin şifrelenmesi Vigenere tablosu kullanılarak da yapılır. Tablonun her satırında 26 Sezar şifresine karşılık gelen bir önceki satıra göre sola kaydırılarak yeni bir alfabe ortaya çıkarılmıştır. Vigenere şifrelemede her farklı harf için farklı bir Sezar kaydırması

kullanılır. Örneğin 2'li Sezar şifrelemesinde 'A' yerine 'C' gelirken 4'lü Sezar şifrelemede 'A' yerine 'E' gelmektedir. Bu sayede şifrede farklı alfabeler kullanılmış olmaktadır (Polyalphabetic).

Sonuç olarak, anlaşıldığı üzere Vigenere şifreleme metodunda bir harfe karşılık birden fazla harf gelebilmektedir. Buradan şu çıkarım yapılabilir, seçilen anahtar ne kadar uzun olursa herhangi bir harfe karşılık gelen harf sayısı da artar dolayısıyla kırılma ihtimali zorlaşır.

Biz bu projemizde Vigenere Cipher Algoritmasını kullanarak Şifreleme/Şifre Çözme uygulaması yazdık.

Ceaser Cipher - Brute Force:

Bu projede ilk şifreleme tekniği olarak kabul edilen Sezar Şifreleme Algoritmasını yazdık. Sezar savaş döneminde önemli bilgilleri bu şifreleme tekniği ile yollarmış. Bu şifreleme tekniği şöyle işlemekte. Sizin şifrelenmesini istediğiniz bir kelime ya da veri mevcut diyelim. Bizim bir anahtar değer belirlememiz gerekmekte. Bu anahtar değeri bir rakam olmalı. Bu anahtar değeri kadar verilen kelimedeki harfleri öteliyoruz. Örneğin, anahtar değeri:3. Kelimedeki harfler 3 sonrasındaki harflere ötelenir ve "deneme" yerine "ghqhph"yazılır.

Biz bu projemizde başta bir Sezar Şifreleme Algoritması ve Brute Force Attack kodu yazdık. Bu yazdığımız Sezar Şifreleme Algoritmasını kullanarak bir metini şifreledik. Daha sonrasında Brute Force Attack kodumuzu kullanarak Sezar Şifreleme Algoritması ile şifrelenmiş bir metini ya da veriyi çözdük.

Figure 2: Brute Force Attack

Proje Tanımı

Projenin İçerdiği İşlevler:

- Mesaj şifreleyebilme (Encrypt)
- Mesaj şifre çözebilme (Decrypt)
- Mesajı kaba kuvvet (Brute Force) ile kırabilme

Kullanıcı bir menüden yapmak istediği işlemi seçiyor ve program ona göre ilerliyor.

```
while True:
    mode = ChooseAlgorithm()
    if mode[0] == 'q':
        print("Exiting to the program...")
        time.sleep(1)
        print("Logged out..")
        break
    if mode[0] == 'v':
        VigenereLoop()
        time.sleep(1)
        continue
    if mode[0] == 'c':
        CeaserLoop()
        time.sleep(1)
        continue
```

Figure 3: Loop

```
def ChooseAlgorithm():
    while True:
        print("--------------------------------------")
        print("--      VIGENERE CIPHER  --> V      --")
        print("--      CEASER CIPHER    --> C      --")
        print("--      QUIT             --> Q      --")
        print("--------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'vigenere v ceaser c quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check..\n")
```

Figure 4: Choose Algorithm()

1. **Kullanıcının Vigenere Cipher Algoritasını seçmesi durumunda:**

   - Kullanıcıyı Vigenere Cipher menüsüne yönlendirir.
   - **Kullanıcının mesaj şifreleme işlemini seçmesi durumunda:**
     - Kullanıcıdan şifreleme anahtarı isteniyor.
     - Kullanıcıdan şifrelenecek metin isteniyor.
     - Metin Vigenere Cipher Encrypt Algoritması ile şifrelenip şifreli mesajı ekrana basıyor.
   - **Kullanıcının mesaj şifre çözme işlemini seçmesi durumunda:**
     - Kullanıcıdan şifreleme anahtarı isteniyor.
     - Kullanıcıdan şifrelenmiş metin isteniyor..
     - Metin Vigenere Cipher Decrypt Algoritması ile deşifre edilip düzgün metni ekrana basıyor.

```
def getModeVigenere():
    while True:
        print("----------------------------------------")
        print("--         VIGENERE CIPHER            --")
        print("----------------------------------------")
        print("-----------     MENU      ------------")
        print("--                                    --")
        print("--             Encrypt       ---> E --")
        print("--             Decrypt       ---> D --")
        print("--             Menu          ---> Q --")
        print("--                                    --")
        print("----------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'encrypt e decrypt d brute b quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check.. \n")
```

Figure 5: getModeVigenere()

2. **Kullanıcının Ceaser Cipher Algoritasını seçmesi durumunda:**
   - Kullanıcıdan şifrelenecek metin isteniyor.
   - Sezar Şifreleme Algoritmasının uygulanması için bir anahtar (key) değer isteniyor.
   - Metin Sezar Şifreleme Algoritmasına girerek şifreleniyor.
   - **Kullanıcının mesaj şifre çözme işlemini seçmesi durumunda:**
     - Kullanıcıdan şifresini çözmek istediği metin isteniyor.
     - Metinin Sezar Şifreleme Algoritması esnasında girilen anahtar (key) değeri isteniyor.
     - Metin Sezar Şifreleme Algoritmasından geçerek şifresi çözülüyor.
   - **Kullanıcının kaba kuvvet işlemini seçmesi durumunda:**
     - Kullanıcıdan kaba kuvvet ile kırmak istediği metin isteniyor.
     - Metin Brute Force Attack Kodu ile tüm ihtimaller denenerek kırılıyor.

```
def getModeCeaser():
    while True:
        print("-----------     MENU      ------------")
        print("--                                   --")
        print("--          Encrypt          ---> E --")
        print("--          Decrypt          ---> D --")
        print("--          Brute Force      ---> B --")
        print("--          Menü             ---> Q --")
        print("--                                   --")
        print("---------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'encrypt e decrypt d brute b quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check..\n")
```

Figure 6: getModeCeaser()

Yöntemler ve Algoritmalar

Kullanılan Algoritma

- Vigenere Cipher Algorithm
- Ceaser Cipher Algorithm

Kullanılan Yöntem

- Encrypt (Şifreleme)
- Decrypt (Deşifreleme)
- Brute Force (Kaba Kuvvet)

Neden Vigenere Cipher Algoritması ?

- Diğer algoritmalara göre daha zor kırılabilen bir algoritma.
- Sezar şifrelemesinin geliştirlmiş halidir.
- Vigenere şifrelemesinde birden fazla alfabe kullanılır.
- Formatı gizli değildir.
- Alıcının ve gönderenin kullandığı anahtar aynıdır.

Neden Sezar Şifreleme Algoritması ?

- Sezar Şifreleme Algoritmasının basitliği yüzünden brute-force saldırısı ile kolayca çözülebilir.
- Formatı gizli değildir.
- Alıcının ve gönderenin kullandığı anahtar aynıdır.

Neden Brute Force Yöntemi ?

- Toplam 26 harf ile işlem yapıldığından en fazla 25 deneme çözmek için yeterli olacaktır.
- Kesin bir sonuç ile şifre kırılacaktır.

Uygulanan Diğer Projeler

Benzerlikler ve Farklılıklar

Vigenere Cipher vs Ceaser Cipher:

Sezar savaş döneminde önemli bilgilleri bu şifreleme tekniği ile yollarmış. Bu şifreleme tekniği şöyle işlemekte. Sizin şifrelenmesini istediğiniz bir kelime ya da veri mevcut diyelim. Bizim bir anahtar değer belirlememiz gerekmekte. Bu anahtar değeri bir rakam olmalı. Bu anahtar değeri kadar verilen kelimedeki harfleri öteliyoruz. Örneğin, anahtar değeri:3. Kelimedeki harfler 3 sonrasındaki harflere ötelenir ve "deneme" yerine "ghqhph"yazılır. Bu anahtar değeri bütün karakterler için geçerlidir ve bütün karakterler 3 harf sonraki harfe ötelenir.

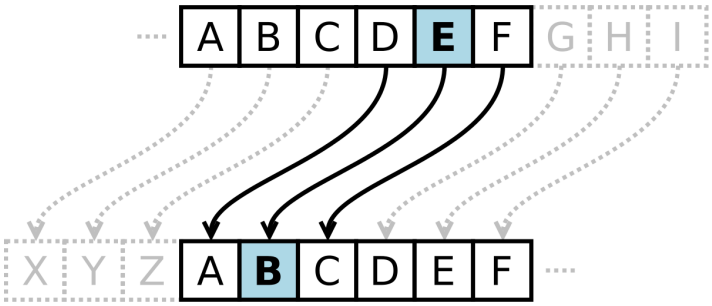


Figure 7: Ceaser Cipher

Bizim projemizden farklı olarak Vigenere Şifrelemede bir harf sürekli belirli bir miktar ötelenmiyor. Anahtar değeri olarak tek bir değer almıyoruz. Metnin uzunluğunca bir anahtar değerimiz olabiliyor. 'AMA' kelimesindeki 2 'A' harfi de farklı harfler olarak

ötelenebilir, aynı olmak zorunda değiller. Bu da Sezar Şifrelemeye göre daha zor kırılabilmesi demektir.

Bizim projemiz ile benzerliği ise, bir key değeri alıp bu değer kadar öteleme yapmasıdır. Kullanıcıdan bir metin ve anahtar değeri alması bir benzerliktir. İki algoritma da şifreleme ve deşifreleme yapmaktadır.

Ceaser Cipher (Brute Force) vs Hydra

Brute Force genellikle internet sitelerinde kullanıcı adı ve şifre kırmak için kullanılıyor.

Bizim projemizden farklı olarak bunun için milyarlarca şifrenin bulunduğu .txt dosyaları veya veritabanları kullanılıyor. Kullanıcı adı ve şifre bulunuyor. Kullanıcı adı ve şifre eşleştiği zaman brute force başarılı bir şekilde uygulanmış, çalışmış oluyor.

Bizim projemiz ile benzerliği ise, deneme yanılma yöntemi. Girilen metin Sezar Şifreleme Algoritması ile şifrelendiği için 25 denemede mesajı kırıp, deşifre edebiliyoruz.

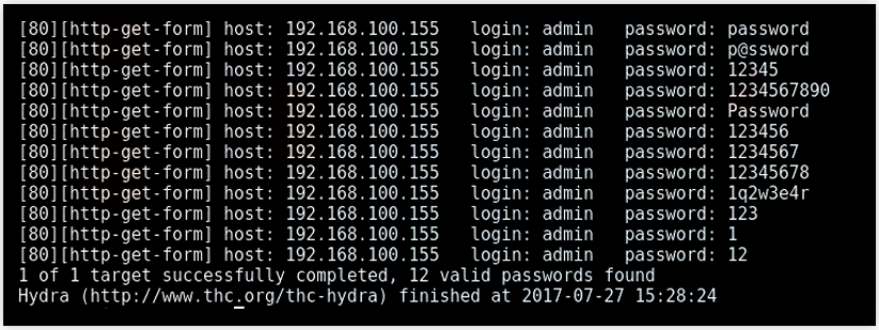


Figure 8: Brute Force - Hydra

Çalışmalarımız

1 Menü Döngüsü

ChooseAlgorithm fonksiyonunu çağırıp kullanıcıyı bir menüye yönlendiriyoruz. Kullanıcının yapmak istediği işlemi seçmesini istiyoruz ve programımız buna göre şekilleniyor.

```
while True:
    mode = ChooseAlgorithm()
    if mode[0] == 'q':
        print("Exiting to the program...")
        time.sleep(1)
        print("Logged out..")
        break
    if mode[0] == 'v':
        VigenereLoop()
        time.sleep(1)
        continue
    if mode[0] == 'c':
        CeaserLoop()
        time.sleep(1)
        continue
```

Figure 9: ChooseAlgorithm Fonksiyonu

2 ChooseAlgorithm Fonksiyonu

Kullanıcıdan hangi algoritma için işlem yapmak istediğini alıyoruz ve kullanıcıyı o algoritmanın menüsüne yönlendiriyoruz.

```
def ChooseAlgorithm():
    while True:
        print("--------------------------------------")
        print("--      VIGENERE CIPHER  --> V      --")
        print("--      CEASER CIPHER    --> C      --")
        print("--      QUIT             --> Q      --")
        print("--------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'vigenere v ceaser c quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check..\n")
```

Figure 10: ChooseAlgorithm Fonksiyonu

3 getModeVigenere Fonksiyonu

Kullanıcıdan Vigenere Cipher ile yapmak istediği işlemi (encrypt/decrypt/menu) bu kısımda alıyoruz.

```
def getModeVigenere():
    while True:
        print("---------------------------------------")
        print("--         VIGENERE CIPHER           --")
        print("---------------------------------------")
        print("-----------     MENU      ------------")
        print("--                                   --")
        print("--             Encrypt       ---> E --")
        print("--             Decrypt       ---> D --")
        print("--             Menu          ---> Q --")
        print("--                                   --")
        print("---------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'encrypt e decrypt d brute b quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check.. \n")
```

Figure 11: getModeVigenere Fonksiyonu

4 VigenereLoop Fonksiyonu

Bu kısımda kontrol yapılarımız yer alıyor. Kullanıcının seçtiği işleme göre fonksiyonlar çalışıyor.

```
def VigenereLoop():
    while True:
        mode = getModeVigenere()
        if mode[0] == 'q':
            print("Returning to the menu...")
            time.sleep(1)
            print("Choose function :  ")
            break
        if mode[0] == 'e':
            print(vigenereEnc())
            time.sleep(1)
            continue
        if mode[0] == 'd':
            print(vigenereDec())
            time.sleep(1)
            continue
```

Figure 12: VigenereLoop Fonksiyonu

5 vigenereEnc Fonksiyonu

Vigenere şifresinde anahtar, bir kelime veya cümledir. Vigenere şifrelemenin algoritmik ifadesi, şifrelenecek mesajın modüler eklenmesi ve anahtar kelimenin tekrarıdır. Şifreli mesajı çözme mantığı da benzerdir, şifreli mesajın ve anahtar kelimenin modüler çıkarılmasıdır.

```
def vigenereEnc():
    alphabet = "abcdefghijklmnopqrstuvwxyz"
    input_string = ""
    enc_key = ""
    enc_string = ""

    enc_key = input("Please enter encryption key:  ")
    enc_key = enc_key.lower()

    input_string = input("Please enter a string of text: ")
    input_string = input_string.lower()

    string_length = len(input_string)

    expanded_key = enc_key
    expanded_key_length = len(expanded_key)

    while expanded_key_length < string_length:
        expanded_key = expanded_key + enc_key
        expanded_key_length = len(expanded_key)

    key_position = 0

    for letter in input_string:
        if letter in alphabet:
            position = alphabet.find(letter)
            key_character = expanded_key[key_position]
            key_character_position = alphabet.find(key_character)
            key_position = key_position + 1
            new_position = position + key_character_position
            if new_position > 26:
                new_position = new_position - 26
            new_character = alphabet[new_position]
            enc_string = enc_string + new_character
        else:
            enc_string = enc_string + letter
    return (enc_string)
```

Figure 13: vigenereEnc Fonksiyonu

6 vigenereDec Fonksiyonu

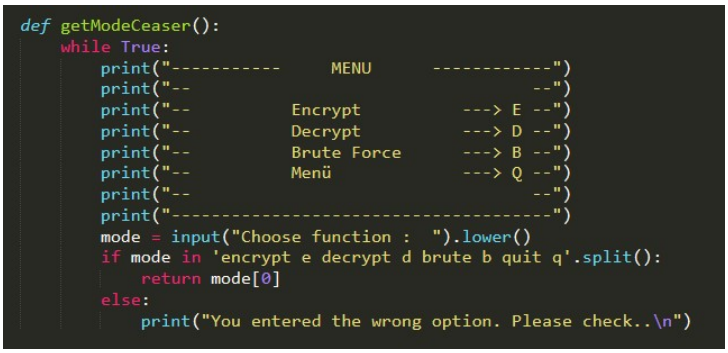
vigenereEnc fonksiyonunda yaptığımız tüm işlemleri geri alarak tekrar kullanıcıdan alınan deşifre edilmiş metni ekrana basıyoruz.

```
def vigenereDec():
    alphabet = "abcdefghijklmnopqrstuvwxyz"
    input_string = ""
    dec_key = ""
    dec_string = ""

    dec_key = input("Please enter encryption key: ")
    dec_key = dec_key.lower()

    input_string = input("Please enter a string of text: ")
    input_string = input_string.lower()

    string_length = len(input_string)

    expanded_key = dec_key
    expanded_key_length = len(expanded_key)

    while expanded_key_length < string_length:
        expanded_key = expanded_key + dec_key
        expanded_key_length = len(expanded_key)

    key_position = 0

    for letter in input_string:
        if letter in alphabet:
            position = alphabet.find(letter)
            key_character = expanded_key[key_position]
            key_character_position = alphabet.find(key_character)
            key_position = key_position + 1
            new_position = position - key_character_position
            if new_position > 26:
                new_position = new_position + 26
            new_character = alphabet[new_position]
            dec_string = dec_string + new_character
        else:
            dec_string = dec_string + letter
    return (dec_string)
```

Figure 14: vigenereDec Fonksiyonu

7 getModeCeaser Fonksiyonu

Kullanıcıdan Ceaser Cipher ile yapmak istediği işlemi (encrypt/decrypt/brute-force/menu) bu kısımda alıyoruz.



```
def getModeCeaser():
    while True:
        print("-----------     MENU      ------------")
        print("--                                   --")
        print("--          Encrypt          ---> E --")
        print("--          Decrypt          ---> D --")
        print("--          Brute Force      ---> B --")
        print("--          Menü             ---> Q --")
        print("--                                   --")
        print("---------------------------------------")
        mode = input("Choose function :  ").lower()
        if mode in 'encrypt e decrypt d brute b quit q'.split():
            return mode[0]
        else:
            print("You entered the wrong option. Please check..\n")
```

Figure 15: getModeCeaser Fonksiyonu

8 CeaserLoop Fonksiyonu

Bu kısımda kontrol yapılarımız yer alıyor. Kullanıcının seçtiği işleme göre fonksiyonlar çalışıyor.

```
def CeaserLoop():
    while True:
        mode = getModeCeaser()
        if mode[0] == 'q':
            print("Returning to the menu...")
            time.sleep(1)
            print("Choose function:  ")
            break
        message = getMessage()
        if mode[0] == 'e':
            key = getKey()
            print(getTranslatedMessage(mode, message, key))
            time.sleep(1)
            continue
        if mode[0] == 'd':
            key = getKey()
            print(getTranslatedMessage(mode, message, key))
            time.sleep(1)
            continue
        if mode[0] == 'b':
            key = 0
            for key in range(1, MAX_KEY_SIZE + 1):
                print(key," ---> ", getTranslatedMessage('decrypt', message, key))
            time.sleep(1)
            continue
```

Figure 16: CeaserLoop Fonksiyonu

9 GetKey Fonksiyonu

Anahtarı kullanıcıdan getKey fonksiyonu ile alıyoruz. Fonksiyonumuzu sadece 1 ve 26 aralığında çalışacak şekilde yazıyoruz.

```
def getKey():
    while True:
        print('Anahtar numarasını girin (1-% s)' % (MAX_KEY_SIZE))
        key = int(input())
        if (key >= 1 and key <= MAX_KEY_SIZE):
            return key
```

Figure 17: getKey Fonksiyonu

10 getMessage Fonksiyonu

Bu fonksiyon içerisinde kullanıdan bir input olarak metin alıyoruz.

```
def getMessage():
    print('Mesajınızı girin:')
    return input()
```

Figure 18: getMessage Fonksiyonu

11 GetTranslatedMassage Fonksiyonu

```
def getTranslatedMessage(mode, message, key):
    if mode[0] == 'd':
        key = -key
    translated = ''
    for symbol in message:
        if symbol.isalpha():
            num = ord(symbol)
            num += key
            if symbol.isupper():
                if num > ord('Z'):
                    num -= 26
                elif num < ord('A'):
                    num += 26
            elif symbol.islower():
                if num > ord('z'):
                    num -= 26
                elif num < ord('a'):
                    num += 26
            translated += chr(num)
        else:
            translated += symbol
    return translated
```

Figure 19: GetTranslatedMassage Fonksiyonu

Mode değişkenindeki ilk harfin 'd' dizesi olup olmadığını kontrol ediyoruz. Eğer öyle ise program şifre çözme modundadır.

.isalpha() ile dizeye sadece harf girilmesini sağlıyoruz.

ord() ile karakterlerin ASCII karşılıklarını alıyoruz.

.isupper() ile metindeki büyük harf karakterleri tespit ediyoruz.

.islower() ile metindeki küçük harf karakterleri tespit ediyoruz.

Eğer kod 'else' şartına girerse program şifreleme moduna giriyor. Girilen anahtar değeri kadar harfleri iteliyor ve şifreliyor.

Sonuç

1. Menü Sonucu

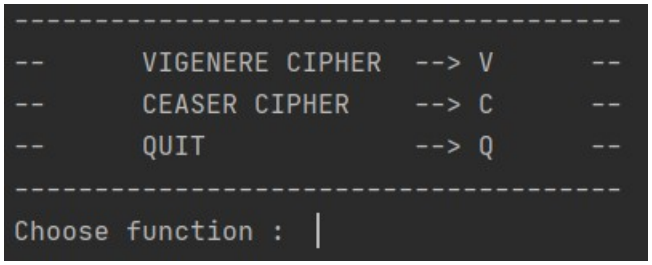


Figure 20: Menü

2. Vigenere Menü Sonucu

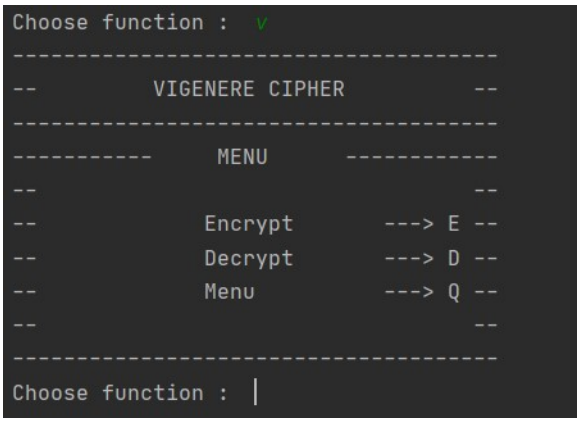


Figure 21: Vigenere Menü

3. Vigenere Şifreleme (Encrypt) Sonucu

```
Choose function :  e
Please enter encryption key:  abcd
Please enter a string of text: ahmad
aiodd
```

Figure 22: Vigenere Şifreleme

4. Vigenere Deşifreleme (Decrypt) Sonucu

```
Choose function :  d
Please enter encryption key: abcd
Please enter a string of text: aiodd
ahmad
```

Figure 23: Vigenere Deşifreleme

5. Ana Menüye Dönme Sonucu

```
Choose function :  q
Returning to the menu...
Choose function :
-------------------------------------
--      VIGENERE CIPHER  --> V      --
--      CEASER CIPHER    --> C      --
--      QUIT             --> Q      --
-------------------------------------
```

Figure 24: Ana Menüye Dönme

6. Ceaser Şifreleme (Encrypt) Sonucu

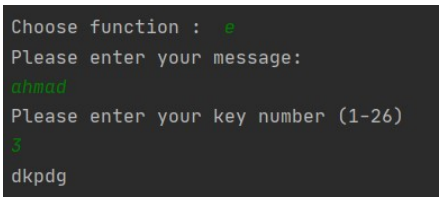


Figure 25: Ceaser Şifreleme

7. Ceaser Deşifreleme (Decrypt) Sonucu

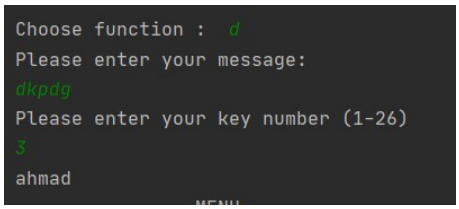


Figure 26: Ceaser Deşifreleme

8. Kaba Kuvvet (Brute Force) Sonucu

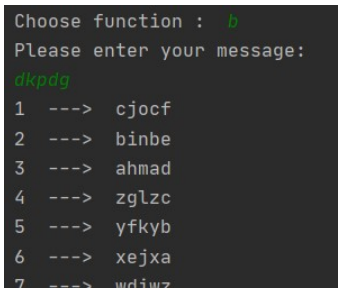


Figure 27: Kaba Kuvvet (Brute Force)

Kaynaklar

NOT: Kaynaklara ulaşmak için üzerine tıklayınız..!

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Vigenere Şifreleme Algoritması

Sezar Şifreleme Algoritması

Sezar Şifreleme Algoritması

Sezar Şifreleme Algoritması

Sezar Şifreleme Algoritması

Sezar Şifreleme Algoritması

Brute Force

Brute Force

Brute Force

Brute Force - Hydra

Brute Force - Hydra