KRİPTOGRAFİ: SEZAR ŞİFRELEME VE ÇÖZME

İÇERİK

- Sezar şifreleme nedir?
- Projenin amacı
- Kullanılan araçlar
- Tasarım
- Sonuçlar
- Proje Sahipleri

SEZAR ŞİFRELEME NEDİR?

- Jül Sezar döneminde savaşta önemli bilgileri iletmek için bu şifreleme tekniğini kullanılmıştır.
- Sezar Şifreleme Yönteminin algoritması aslında oldukça basittir. Ana mesajda bulunan her bir harfi mesajda belirtilen anahtar sayısı kadar ileri götürülerek şifreli mesaj oluşturulabilir.

SEZAR ŞİFRELEME NEDİR?

- Şifreli mesajı okumak için ise mesajda bulunan her harfi anahtar sayısı kadar geriye götürmek gerekir.
- Günümüz de ise **Sezar Şifreleme Yöntemi** pek sık tercih edilmemektedir. Çünkü zamanında güvenli olsa bile şuan şifrenin çözülebilme olasılığı 1/25'dir

PROJENİN AMACI

Projemizin amacı sezar şifreleme tekniği ile hem şifreleme hem de şifre çözmeyi gerçekleştirebilen c dillinde bir kod yazmaktır.

KULLANILAN ARAÇLAR

- Tasarım sürecinde Visual Studio kullanılmıştır. Visual Studio, birçok programlama dilini kullanarak program, uygulama ya da web sitesi yapabileceğiniz bir IDE yani entegre geliştirme ortamıdır. Microsoft Windows için bilgisayar programları, web siteleri, web uygulamaları, web hizmetleri ve mobil uygulamalar geliştirmek için kullanılır.

TASARIM

- Yazdığımız kodda fonksiyon , dosya okuma ve yazma yapılarını kullandık. Biri şifreleme fonksiyonu iken ikincisi şifre çözme fonksiyonudur. Her iki fonksiyonda da büyük veya küçük harf olmasına dikkat ederek farklı işlemler gerçekleştirdik.
- Şifreleme yapılırken şifrelenecek bilgi giris.txt dosyasından okunup sifreli.txt dosyasına yazılacaktır. Şifre çözülürken bilgi sifreli.txt dosyasından okunup cıkıs.txt dosyasına yazılacaktır.

- Main fonksiyonumuzda öncelikle kullanıcıdan hangi işlemi yapmak istediğini belirten sayıyı aldık. Şifreleme işlemi için 1 sayısını, şifre çözme işlemi için 2 sayısını belirledik.
- Şekil 1 de şifreleme seçildiği zaman yapılan işlemin gerçekleştirildiği kod parçasıdır. Öncelikle kullanıcıdan kaç karaktere göre şifrelenmesini istediği yani anahtar sayı alınır. Sonrasında giriş.txt dosyasını okuma işlemi gerçekleştirilir ve daha sonra şifrele fonksiyonu çağırılır. Şifreleme işlemi gerçekleştirildikten sonra şifrelenen cümleler sifreli.txt adlı dosyaya yazılır. Böylelikle şifreleme işlemi gerçekleştirilmiş olur.

```c
int main(void)
{
    FILE* dosya;
    FILE* dosya2;
    char cumle[250];
    int menuSecim, sifrelenicekKarakter;
    printf("Lutfen islem turunu seciniz:\nSifrelemek icin ->1\nSifre Cozmek icin ->2");
    scanf_s("%d", &menuSecim);
    if (menuSecim == 1) {

        int rtn = fopen_s(&dosya, "giris.txt", "r");
        fopen_s(&dosya2, "sifreli.txt", "w");
        if (rtn != 0) {
            return 0;
        }
        printf("Kac karaktere gore sifrelenmesini istersiniz ? :");
        scanf_s("%d", &sifrelenicekKarakter);
        while (fscanf_s(dosya, "%[^\n]", cumle,250) != EOF)
        {

            sifrele(cumle, sifrelenicekKarakter);


            fprintf(dosya2, "%s", cumle);

        }
        fclose(dosya);
        fclose(dosya2);


    }
```

Şekil 1

- Şekil 2 deki kod parçası şifreleme fonksiyonunda büyük harflere yapılacak olan işlemi göstermektedir.

- Öncelikle boşluk karakterinin şifrelenmemesi için ayrı bir işlem yaptırdık. Sonrasında dosyadan okunan her harfi tek tek kullanıcıdan alınan anahtar sayıya göre şekil 2 deki işlem yaptırılarak şifrelenmiş haline getirdik ve cumle[] dizisi yerine şifrelenmiş cümleyi yazdırdık. Böylelikle şifreleme işi tamamlanmış oldu.

```
int sifrele(char cumle[], int sifrelenicekKarakter) {
    if ((int)cumle[i] > 64 && (int)cumle[i] < 91) {
        for (i = 0; cumle[i]; i++) {
            if (cumle[i] == ' ')
            {
                cumle[i] == ' ';
            }
            else
            {
                char b;
                b = (char)(((int)cumle[i] - 65 + sifrelenicekKarakter) % 26 + 65);
                cumle[i] = b;
            }


        }

    }
```

Şekil 2

- Şekil 3 deki kod parçası şifreleme fonksiyonunda küçük harflere yapılacak olan işlemi göstermektedir. Öncelikle boşluk karakterinin şifrelenmemesi için ayrı bir işlem yaptırdık. Sonrasında dosyadan okunan her harfi tek tek kullanıcıdan alınan anahtar sayıya göre şekil 3 deki işlem yaptırılarak şifrelenmiş haline getirdik ve cumle[] dizisi yerine şifrelenmiş cümleyi yazdırdık. Böylelikle şifreleme işi tamamlanmış oldu.

```
else if ((int)cumle[i] > 96 && (int)cumle[i] < 123) {
    for (i = 0; cumle[i]; i++) {
        if (cumle[i] == ' ')
        {
            cumle[i] == ' ';
        }
        else {
            char b;
            b = (char)(((int)cumle[i] - 97 + sifrelenicekKarakter) % 26 + 97);
            cumle[i] = b;

        }


    }

}
```

Şekil 3

- Şekil 4 deki kod parçası şifre çözme seçildiği zaman yapılan işlemin gerçekleştirildiği kod parçasıdır. Öncelikle kullanıcıdan kaç karaktere göre şifrenin çözülmesinin istediği yani anahtar sayı alınır. Sonrasında sifreli.txt dosyasını okuma işlemi gerçekleştirilir ve daha sonra şifrele fonksiyonu çağırılır. Şifre çözme işlemi gerçekleştirildikten sonra şifrelenen cümleler cikis.txt adlı dosyaya yazılır. Böylelikle şifre çözme işlemi gerçekleştirilmiş olur.

```
else if (menuSecim == 2) {
    fopen_s(&dosya2, "sifreli.txt", "r");
    int rtn = fopen_s(&dosya, "cikis.txt", "w");
    if (rtn != 0) {
        return 0;
    }
    while (fscanf_s(dosya2, "%[^\n]", cumle, 250) != EOF)
    {
        printf("Kac karaktere gore cozulmesini istersiniz ? :");
        scanf_s("%d", &sifrelenicekKarakter);
        sifrecoz(cumle, sifrelenicekKarakter);


        fprintf(dosya, "%s", cumle);

    }
    fclose(dosya2);
    fclose(dosya);


}
```

Şekil 4

- Şifre çözme işlemini yapan fonksiyonu yazarken öncelikle şifresi çözülecek karakterin büyük veya küçük harf olmasına dikkat ederek farklı işlemler yaptırdık. Şekil 5 deki kod parçası şifreçöz fonksiyonunda büyük harflere yapılacak olan işlemi göstermektedir.
- Öncelikle boşluk karakterinin şifresinin çözülmemesi için ayrı bir işlem yaptırdık. Sonrasında dosyadan okunan her harfi tek tek kullanıcıdan alınan anahtar sayıya göre şekil 5 deki işlem yaptırılarak şifresini çözdürdük ve cumle[] dizisi yerine şifresi çözülmüş cümleyi yazdırdık. Böylelikle şifre çözme işi tamamlanmış oldu.

```
int sifrecoz(char cumle[], int sifrelenicekKarakter) {


    if ((int)cumle[i] > 64 && (int)cumle[i] < 91) {
        for (i = 0; cumle[i]; i++) {

            if (cumle[i] == ' ')
            {
                cumle[i] = ' ';
            }
            else
            {

                char b;
                b = (char)(((int)cumle[i] - 65 - sifrelenicekKarakter % 26 + 26) % 26) + 65;
                cumle[i] = b;
            }

        }

    }
```

Şekil 5

- Şekil 6 deki kod parçası şifreçöz fonksiyonunda küçük harflere yapılacak olan işlemi göstermektedir.
- Öncelikle boşluk karakterinin şifresinin çözülmemesi için ayrı bir işlem yaptırdık. Sonrasında dosyadan okunan her harfi tek tek kullanıcıdan alınan anahtar sayıya göre şekil 6 deki işlem yaptırılarak şifresini çözdürdük ve cumle[] dizisi yerine şifresi çözülmüş cümleyi yazdırdık. Böylelikle şifre çözme işi tamamlanmış oldu.
-

```
else if ((int)cumle[i] > 96 && (int)cumle[i] < 123) {
    for (i = 0; cumle[i]; i++) {

        if (cumle[i] == ' ')
        {
            cumle[i] = ' ';
        }
        else {
            char b;


                b = (char)(((int)cumle[i] - 97 - sifrelenicekKarakter % 26 + 26) % 26) + 97;

                cumle[i] = b;


        }


    }

}
```

Şekil 6

- Şekil 7 de kullanıcıdan hangi işlemi gerçekleştirmek istediği sorulduğunda 1 veya 2 sayısını girilmezse kullanıcını hatalı seçim yaptığını belirten kod parçası gösterilmiştir.

```
else
{
    printf("Hatali secim.");
}
```

Şekil 7

SONUÇLAR

- Proje sonucunda Sezar Şifreleme ve şifreleme kodu yazma hakkında bilgi sahibi olduk. Ayrıca birden fazla kişiyle kod yazmayı ve bir kodla günlerce uğraşıp geliştirmeyi öğrenmiş olduk. İleride yapacağımız projelerde nelerle karşılaşacağımızla alakalı deneyime sahip olduk.

PROJE SAHİPLERİ

- Aysen İpek Çakır
- İrem Kalkanlı

DİNLEDİĞİNİZ İÇİN TEŞEKKÜRLER