

# PI'RELEME

## İçindekiler

1.Giriş: .....	2
1.1. Projenin Amacı: .....	2
1.2. Windows .....	2
1.3. C#.....	3
2. Yöntem.....	3
2.1. Proje Yapım Basamakları .....	3
a) Pi'releyici(Şifreleyici).....	3
b) DePi'releyici(Deşifreleyici).....	4
3. Bulgular ve Gerçekleşme .....	4
a. Pi'releyicinin Yapımı .....	4
a.1. Basit arayüz oluşturulması .....	4
a.2. Dönüştürücülerin Programlanması.....	4
a.3. Pi sayısının, programın kullanabileceği bir dosya haline getirilmesi .....	4
b. DePi'releyici'nin Yapımı.....	4
b.1. Ana arayüze ek DePi'releyici arayüzünün oluşturulması .....	4
b.2. Dönüştürücülerin Programlanması .....	5
3.4. Proje Kodlarının Açıklanması.....	5
3.4.1. App.config .....	5
3.4.2 Form1.cs (1. Form) .....	5
3.4.3 Form1.Designer.cs (1. Form).....	6
3.4.4. Form1.cs.....	6
3.4.5. Form2dePireleme.Designer.cs (2. Form).....	8
3.4.6. Form2dePireleme.cs (2. Form) .....	9
3.5. Proje Ekran Görüntüleri .....	11
4. Sonuçlar ve Tartışma .....	12
5.Öneriler .....	12
Kaynakça.....	13

**1.Giriş:** Şifreleme genel anlamıyla bir mesajın, o mesajı çözmek için (decrypt) bir anahtara sahip olan kişi veya kişiler haricinde okunmasını engelleyen matematiksel işleme verilen isimdir. İnsanlar, tarih boyunca yalnızca istedikleri kişinin okumasını umdukları mesajları şifreleme yöntemini kullanarak göndermişlerdir.<sup>1</sup> Günümüzde ise bu şifreleme işlemini bizim yerimize yapabilen bilgisayarlara sahibiz. Bugün gizli mesajlaşmanın da ötesine giden dijital şifreleme teknolojisi, mesajların yazarını doğrulamak gibi özel amaçlar için de kullanılabilir. Doğru kullanıldığında kırılması neredeyse imkânsız olan şifreleme teknolojisi, elimizdeki bilgileri kötü aktörlerden, devletlerden ve servis sağlayıcılarından korumamız için günümüzde sahip olduğumuz en önemli araçlardan biri.<sup>2</sup>

Şifrelemedeki genel sorun ise şifreleme anahtarlarının sunucu bazlı olmalarıdır. Bu, verilerin belli bir sunucudan geçmesini ve yine gönderenin bu veri gönderdiği servisin sunucusuna güvenmekten başka çaresinin olmadığı anlamına gelmektedir. Bu sebeple projedeki temel amaçlardan biri de bir sunucu gereksinimini kaldırmak ve şifreleme anahtarını verinin içinde gizleyerek göndermeyi sağlamaktır.<sup>3</sup>

Tüm bunları yaparken pi sayısının kullanımındaki sebep bu sayının dünya genelinde kabul görmüş bir sayı olmasıdır. Ayrıca bu sayının asla kendini tekrar etmeyen bir sayı oluşu da yine bu sayının ne kadar güvenilir olduğunu kanıtlamaktadır.

**1.1. Projenin Amacı:** Deşifre anahtarı kendi içinde olan ve mesaj özelliklerine göre konumlanan bu rastgele-mesaj aynı olsa dahi- anahtarla çalışan bir şifreleme metodu üretmektir. Bu metot pi sayısının kendini asla tekrâr etmeyişiinden faydalanan, her alanda kullanılabilecek bir güvenlik, kolaylık ve hafifliğe sahip güvenilir bir şifreleme algoritması oluşturmak ve bu algoritmayı kullanarak güçlü bir şifreleme metodu oluşturmaktır.

Sonuç olarak; bu metodun önce metin türü verilerde daha sonra ise her türlü(resim, video, mp3 vs...) veride kullanımı ve yine oluşturulan metodun hem uluslararası hem de daha güvenilir olması planlanmaktadır.

## 1.2. Windows

Projede söz konusu olan programımızın içerisinde kullanılan c#(sharp) dilinin getirdiği ağırlıktan dolayı en uyumlu olduğu ve en hızlı derleyen işletim sistemi olan Windows platformu tercih edildi.<sup>4</sup>

Windows'ta uygulama geliştirebilmek için Visual Studio, MonoDevelop vb. gibi bir geliştirme aracına ihtiyaç duyulmaktadır. Projede Visual Studio 2017 Community Edition kullanılmıştır.

---

<sup>1</sup> <https://searchsecurity.techtarget.com/definition/encryption>

<sup>2</sup> <https://ssd.eff.org/tr/module/%C5%9Fifreleme-nedir>

<sup>3</sup> <https://tr.khanacademy.org/computing/computer-science/cryptography/ciphers/a/ciphers-vs-codes>

<sup>4</sup> <https://www.digitalunite.com/technology-guides/computer-basics/using-computer/what-windows>

Visual Studio; Microsoft tarafından 1997 yılında “Boston” kod adı ile yayımlanmıştır.<sup>5</sup> Visual Studio’nun yayımlanan son sürümü 15,9’dur ve 11 Kasım 2018 tarihinde yayımlanmıştır.

### 1.3. C#

C# (si şarp şeklinde telaffuz edilir), Microsoft’un geliştirmiş olduğu yeni nesil programlama dilidir.<sup>6</sup> Yine Microsoft tarafından geliştirilmiş DOTNET Teknolojisi için geliştirilmiş dillerden biridir. Microsoft tarafından geliştirilmiş olsa da ECMA ve ISO standartları altına alınmıştır.<sup>7</sup>

ECMA tarafından C# dilinin tasarım hedefleri aşağıdaki şekilde sıralanır:

- C# modern, basit, genel-amaçlı, object oriented(nesneye yönelik) programlama dili olarak tasarlanmıştır.
- Programcı portatifliğine sahip bir dil olması ile özellikle C ve C++ dilleri ile tecrübesi olanlar için çok önemlidir.
- C# programlama dili sunucu ve gömülü sistemler için tasarlanmıştır.
- C# programlama dili en hafif işleve sahip fonksiyondan, işletim sistemini kullanan en ayrıntılı fonksiyonu dahi kapsamaktadır.
- C# uygulamaları hafıza ve işlemci gereksinimleri ile tutumlu olmak üzere tasarlanmıştır. Buna rağmen C# programlama dili performans açısından C veya assembly dili ile rekabet etmek için tasarlanmamıştır.
- DOTNET Microsoft uygulama bonservisi Windows üzerinde geçerlidir. Fakat C# programlarını Windows, Linux veya MacOS üzerinde yürüten başka uygulamalar da yer almaktadır. MonoSoftware, DotGnu, Wine vb. .

## 2. Yöntem

Projenin Yapımı, üç temel aşamada planlanmıştır. İlk aşama pi sayısı kullanılarak projenin şifreleme algoritmasının tasarlanması ve bir Windows Forms App’e(Windows Formlar Uygulaması) uyarlanmasıdır. İkinci aşama İki bölümden oluşacak Windows Forms App’in birinci bölümü olan encryptor(şifreleyici) kısmının oluşturulmasıdır. Üçüncü aşama ise Windows Forms App’in ikinci bölümü olan decryptor(şifre çözücü) kısmının oluşturulmasıdır.

### 2.1. Proje Yapım Basamakları

#### a) Pi’ releyici(Şifreleyici)

a.1. Basit bir arayüz oluşturulması

a.2. Metin türünden girilen tüm verilerin, bu proje için özel olarak belirlenen her karakterin sayısal karşılıklarının bulunduğu char to integer(karakter-karakterin sayısal karşılığı) tablosundaki değerlere dönüştürülmesi

---

<sup>5</sup> <https://docs.microsoft.com/en-us/visualstudio/releases/notes/vs2017-relnotes/>

<sup>6</sup> <http://www.csharpnedir.com/>

<sup>7</sup> <https://gelecegiyazanlar.turkcell.com.tr/konu/c-sharp/egitim/c-101/c-dilinin-tarihcesi>

a.3. Pi sayısından her veriye özel rastgele uzunlukta olan rastgele bir bölümünün seçilip, verinin özelliğine göre belli matematiksel işlemlerden sonra sonucun pi kodu olarak kullanıcıya verilmesi.

a.4. Pi sayısının hangi kısmının kullanıldığı ve basamak uzunluğu girilen veri türünün özelliklerine göre pi kodunun başı ve ya sonuna eklenmesi

#### **b) DePi'releyici(Deşifreleyici)**

b.1. Şifreleyici arayüzüne ek, basit bir arayüze sahip yeni bir Windows formu oluşturulması.

b.2. Pi kodu son ve ya baş kısmından okunarak hangi bölümün ne uzunlukta kullanıldığı tespit edilip buna göre şifreleme aşamasında yapılan matematiksel işlemlerin tersinden yapılıp bu proje için özel olarak belirlenen her karakterin sayısal karşılıklarının bulunduğu integer to inttochar(sayı-karakterin sayısal karşılığı) tablosundaki değerlere dönüştürülmesi.

b.3. Elde edilen sayı karşılıklarının bu proje için özel olarak belirlenen her karakterin sayısal karşılıklarının bulunduğu inttochar to char (karakterin sayısal karşılığı-karakter) tablosundaki değerlere dönüştürülmesi.

### **3. Bulgular ve Gerçekleşme**

#### **a. Pi'releyicinin Yapımı**

##### **a.1. Basit arayüz oluşturulması**

Bu aşamada kullanıcının metin türündeki verilerini girip gereken dönüşümleri yapmasını sağlayacak arayüz bir taslak olarak tasarlanacak ve programlanacaktır.

##### **a.2. Dönüştürücülerin Programlanması**

Arayüzün arka planında çalışacak ve metin-sayı-pi kodu dönüşümlerini yapacak script programlanacaktır.

##### **a.3. Pi sayısının, programın kullanabileceği bir dosya haline getirilmesi**

pi sayısının ilk bir milyon basamağının <https://www.piday.org/million/> url'li web sitesinden indirilip programın okuyabileceği bir formata(txt) dönüştürülecektir ve programın içine gömülecektir.<sup>8</sup>

#### **b. DePi'releyici'nin Yapımı**

##### **b.1. Ana arayüze ek DePi'releyici arayüzünün oluşturulması**

Temel program arayüzüne ek olarak buton ile ulaşılabilir basit bir arayüz olan DePi'releme arayüzü programlanacaktır.

---

<sup>8</sup> <https://www.piday.org/million/> Pi sayısının ilk bir milyon basamağı

## b.2. Dönüştürücülerin Programlanması

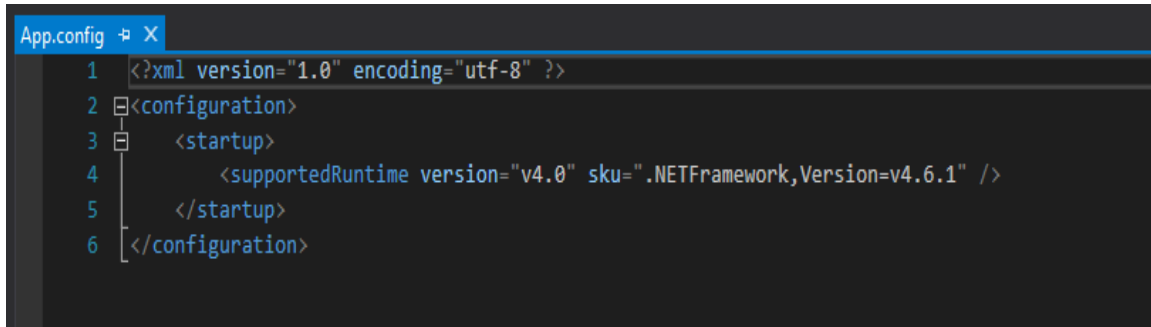
Arayüzün arka planında çalışacak ve pi kodu – sayı - metin dönüşümlerini yapacak script programlanacaktır.

Projede tüm kullanıcıların ulaşabileceği toplam 2 adet Form kullanılmıştır.

## 3.4. Proje Kodlarının Açıklanması

Bu bölümde proje içerisinde kullandığımız kod bloklarından bir kısmı gösterilecek, açıklanacak ve bu kod bloklarının hangi işlemleri gerçekleştirdiğinden bahsedilecektir. Kod blokları resim halinde gösterildikten sonra parça parça ne işe yaradığı anlatılacaktır.

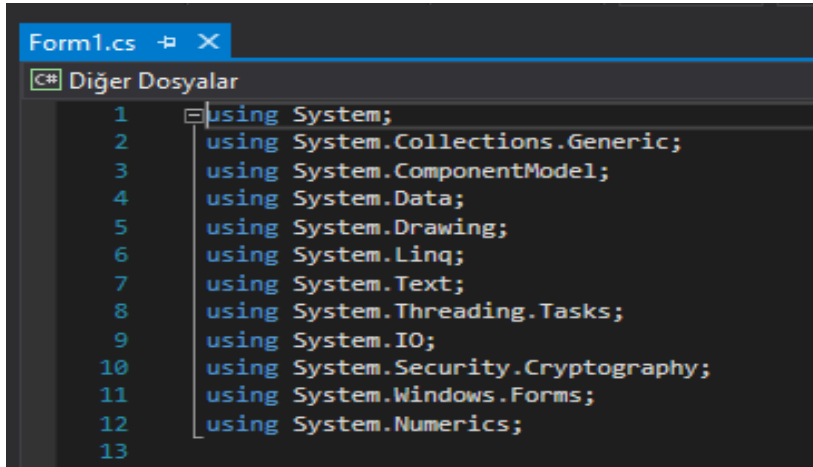
### 3.4.1. App.config



```
App.config  X
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <startup>
4     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
5   </startup>
6 </configuration>
```

Burada programın çalışması için gereken minimum DotNet Sürümü ayarlanmıştır.

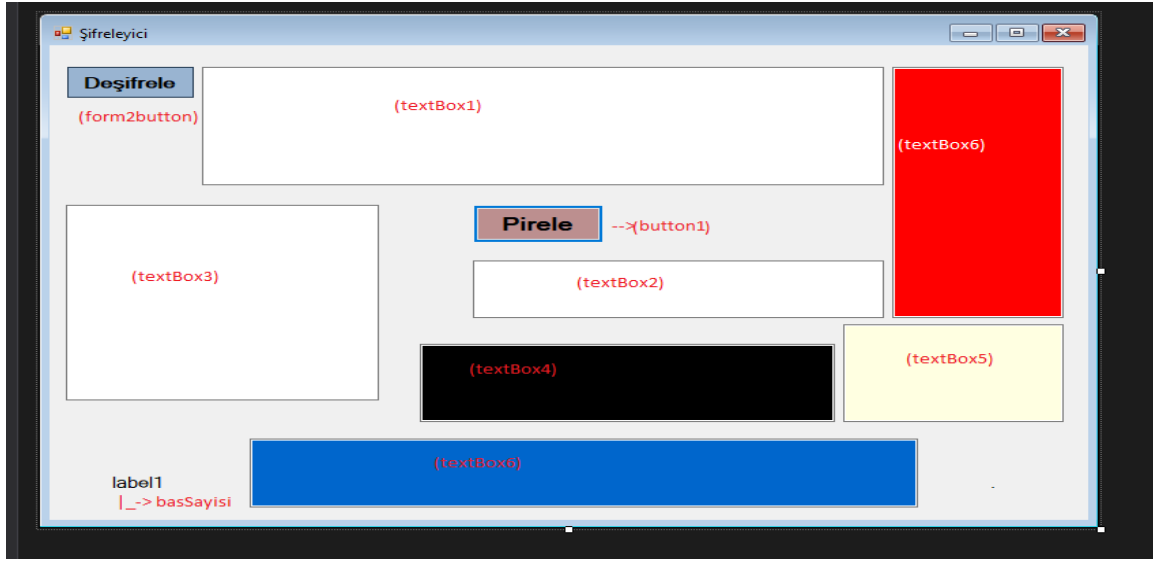
### 3.4.2 Form1.cs (1. Form)



```
Form1.cs  X
C# Diğer Dosyalar
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.IO;
10 using System.Security.Cryptography;
11 using System.Windows.Forms;
12 using System.Numerics;
13
```

Burada programdaki 1. Formun ihtiyacı olan c# kütüphaneleri seçilmiştir.

### 3.4.3 Form1.Designer.cs (1. Form)



Burada form 1 in kodlanırken kullanılacak arayüz öğeleri belirtilmiştir.

### 3.4.4. Form1.cs

```
29 private void button1_Click(object sender, EventArgs e)
30 {
31
32     var cipherInvertal = new Random();
33     var TextToCipher = (string)textBox1.Text;
34     textBox2.Visible = true;
35     textBox2.PasswordChar = '*';
36     textBox1.Text = TextToCipher;
37     textBox2.Enabled = false;
38     textBox2.Text = textBox1.Text;
39     textBox3.Text = textBox2.Text;
40     string value = TextToCipher;
41     string s = TextToCipher;
42     var mesaj = textBox3.Text;
43     if (textBox1.Text.Contains("a"))
```

Burada dönüştürülecek veriler TextToCipher, s gibi değişkenlere atanmış olup orijinali bozulmaması amacı ile farklı textBox'lara taşınmıştır.

```

345 {
346     textBox3.Text = textBox3.Text.Replace("B", "566");
347 }
348
349 if (textBox1.Text.Contains("C"))
350 {
351     textBox3.Text = textBox3.Text.Replace("C", "567");
352 }
353
354 if (textBox1.Text.Contains("D"))
355 {
356     textBox3.Text = textBox3.Text.Replace("D", "468");
357 }
358
359 if (textBox1.Text.Contains("E"))
360 {
361     textBox3.Text = textBox3.Text.Replace("E", "569");
362 }
363
364 if (textBox1.Text.Contains("F"))
365 {
366     textBox3.Text = textBox3.Text.Replace("F", "570");
367 }
368
369 if (textBox1.Text.Contains("G"))
370 {
371     textBox3.Text = textBox3.Text.Replace("G", "571");
372 }
373
374 if (textBox1.Text.Contains("H"))
375 {
376     textBox3.Text = textBox3.Text.Replace("H", "572");
377 }
378
379 if (textBox1.Text.Contains("I"))
380 {
381     textBox3.Text = textBox3.Text.Replace("I", "573");
382 }

```

Burada “If Contains(eğer içeriyor ise)” tekniği kullanılarak karakterler, atanan sayı değerlerine dönüştürülüp textBox3 içerisinde depolanmaktadır.

```

650
651 string[] allLines = File.ReadAllLines(Application.StartupPath+ "\\pi.txt");
652 Random rnd1 = new Random();
653 string ad=(allLines[rnd1.Next(allLines.Length)]);
654
655 byte[] byteArray = new byte[1024];
656 var file = Application.StartupPath + "\\pi.txt";
657 using (BinaryReader reader = new BinaryReader(new FileStream(file, FileMode.Open)))
658 {
659     Random rn = new Random();
660     Random r = new Random();
661     int rInt = r.Next(100000, 999999); //intler için
662     int rnInt = r.Next(512, 999);
663
664     reader.BaseStream.Seek(rInt, SeekOrigin.Begin);
665     reader.Read(byteArray, 0, rnInt);
666
667     kacinciBas.Text = rInt.ToString();
668     basSayisi.Text = rnInt.ToString();
669 }

```

Burada programın içine gömülen “pi.txt” dosyası “allLines” adlı değişkene atanır ve bir C# özelliği olan “BinaryReader” 1024 byte parametresi ile çağrılır. Buradaki “BinaryReader’in”, “pi.txt” dosyasından okuyacağı verilerin rastgele uzunlukta ve rastgele noktalardan başlamasını sağlayan random parametreleri vardır. Bu parametreler “rn” ve “r”dir ve yine bu parametreler “rInt” ve “rnInt” adlı sayı değişkenlerine belli sınırlarla seçilerek atanır. Bu işlemler sonucu pi kodu üretilir ve pi kodunun anahtarı “kacinciBas ve basSayisi textBox”larına atanır. Sonuç olan pi kodu BinaryReader tarafından result adlı değişkene atanır.

```

else
{
    textBox4.Text = result;
    string coz = kacinciBas.Text + basSayisi.Text;
    textBox6.Text = textBox4.Text;
    //textBox5.Text = mesaj;
    // textBox7.Text = (Convert.ToInt32(mesaj) + Convert.ToInt32(textBox6.Text)).ToString();
    //rslt2 = int.Parse(textBox6.Text);
    BigInteger total;
    total = BigInteger.Parse(textBox3.Text) + BigInteger.Parse(textBox6.Text); || total = BigInteger.Parse(textBox3.Text) / BigInteger.Parse(textBox6.Text);
    textBox7.Text = total.ToString() + coz.ToString();
}

```

Burada, üst bölümde atanan result değeri textBox4 e kaydedilir. Daha sonra bu değer textBox3'deki karakter-sayı karşılıkları ile ya toplanır ya da bölünür. Son olarak kullanıcıya teslim edilecek pi kodunun baş veya son kısmına eklenir.(işlemsiz).

```

775 private void Form1_Load(object sender, EventArgs e)
776 {
777     MessageBox.Show("Mesajınızın Uzunluğu, Şifreleme, Deşifreleme İşlemlerinin Uzunluğunu Etkiler.", "Uyarı");

```

Bu işlemlerin tamamının uzunluğu ise girilen verinin boyutuna göre değişiklik göstermektedir. Bu sebeple kullanıcıya, programın donmadığı, şu şekilde bir uyarı yolu ile program her açıldığında popup şeklinde gelecek bir biçimde bildirilmek üzere programlanmıştır. Bu işlemde Windows Forms Apps'in MessageBox özelliği kullanılmıştır.

### 3.4.5. Form2dePireleme.Designer.cs (2. Form)

Burada form1'e bir buton ile eklenen Form2'nin kodlanırken kullanılacak arayüz öğeleri belirtilmiştir.



### 3.4.6. Form2dePireleme.cs (2. Form)

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.IO;
7  using System.Linq;
8  using System.Text;
9  using System.Threading.Tasks;
10 using System.Windows.Forms;
11 using Microsoft.VisualBasic;
12 using System.Numerics;
13
```

Burada programdaki 2. Formun ihtiyacı olan c# kütüphaneleri seçilmiştir.

```
25 private void dePirele_Click(object sender, EventArgs e)
26 {
27
28     if (textBox1.Text.Length < 20)
29     {
30         MessageBox.Show("Pi kodu 50 karakterden kısa olamaz", "Uyarı");
31     }
32
33     else if (textBox1.Text=="")
34     {
35         MessageBox.Show("Lütfen Mesajınızı giriniz","Uyarı");
36     }
37     else
38     {
39         string tumDepireleyici = textBox1.Text;
```

Görselde textBox1'e girilecek pi kodunun yanlış girilmesini ve boş bırakılmasını önlemek amaçlı uyarı mesajları verilmiş olup daha sonra şartlar (pi kodunun yanlış girilmemesi ve alanın boş bırakılmaması) uygun olduğunda "veri depi'releme" işleminin başlatılması sağlanmıştır. Kullanıcıdan alınan pi kodu ise tumDepireleyici adlı değişkene atanmıştır.

```

string tumDepireleyici = textBox1.Text;
tumDepireleyici = tumDepireleyici.Substring(tumDepireleyici.Length - 9); || tumDepireleyici = tumDepireleyici.Substring(tumDepireleyici.Length + 9);
toplamlar.Text = tumDepireleyici.ToString();

string uzunluk = toplamlar.Text;
uzunluk = uzunluk.Substring(uzunluk.Length - 3);
denItibaren.Text = uzunluk.ToString();
string bas = toplamlar.Text;
bas = bas.Substring(0,6);
kacBasamak.Text = bas.ToString();
textBox1.Text = textBox1.Text.Remove(textBox1.Text.Length-9);
// tumDepireleyici = tumDepireleyici.Remove(tumDepireleyici.Length - 9);
toplamlar.Text = tumDepireleyici.ToString();
string[] allLines = File.ReadAllLines(Application.StartupPath + "\\pi.txt");
Random rnd1 = new Random();
string ad = (allLines[rnd1.Next(allLines.Length)]);

byte[] byteArray = new byte[1024];
var file = Application.StartupPath + "\\pi.txt";
using (BinaryReader reader = new BinaryReader(new FileStream(file, FileMode.Open)))
{
    int rInt = int.Parse(kacBasamak.Text);
    int rnInt = int.Parse(denItibaren.Text);
    reader.BaseStream.Seek(rInt, SeekOrigin.Begin);
    reader.Read(byteArray, 0, rnInt);
}
//int rslt2;
string result = System.Text.Encoding.UTF8.GetString(byteArray);
mesaj.Text = result;
BigInteger an = BigInteger.Parse(textBox1.Text) - BigInteger.Parse(mesaj.Text);
BigInteger an = BigInteger.Parse(textBox1.Text) * BigInteger.Parse(mesaj.Text);
sonmesajBinary.Text = an.ToString();
// mesaj.Text = a.ToString();
}

```

Burada, tumDepireleyici değişkenine atanan pi kodu son ve ilk basamaklarından anahtar ve kod olmak üzere iki ye ayrılır. Daha sonra, form1'de de kullanılan BinaryReader fonksiyonu 1024 byte ve okuma konumu olarak programın içine gömülü pi.txt parametreleri ile çağrılır. Pi kodundan ayrıştırılan anahtar da kacBasamak ve denItibaren label'larına atanır. Önceki formda kullanılan BinaryReader okuma başlangıç ve uzunluk noktaları rastgele değerlerden anahtar değerlerine çevrilir. Bunun sonucunda ortaya çıkan result adlı değişkene atanan sonuç, pi kodundan çıkarılır ve ya pi kodu ile çarpılır(işlemler tersine çevrilir.) Ve sonuç olarak elde edilen "karakterlerin sayı karşılığı" da sonmesajBinary adlı textBox'a kaydedilir.

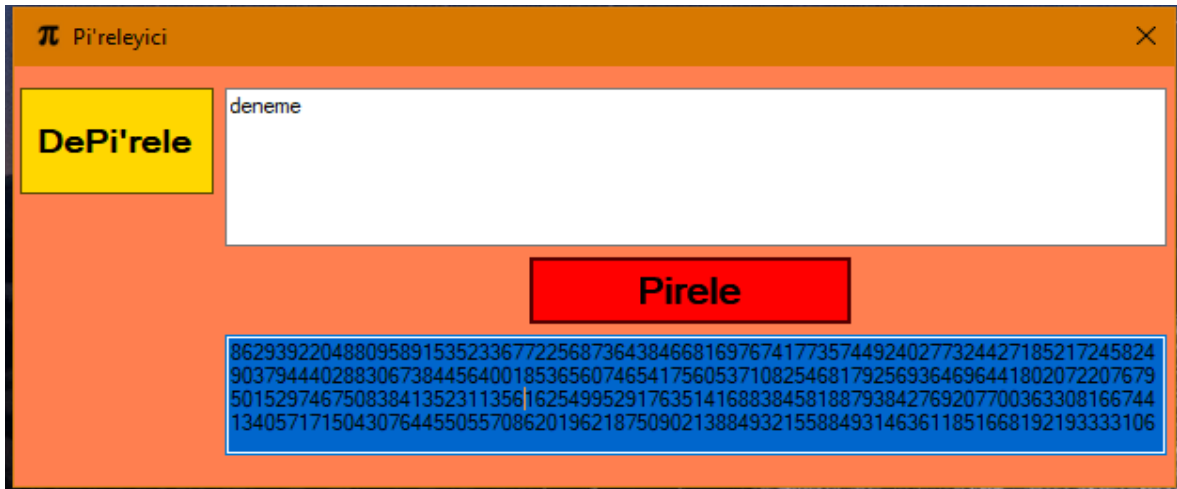
```

76     string a;
77     a = okunanMesaj.Text;
78     if (a.Contains("597"))
79     {
80         sonmesajBinary.Text = sonmesajBinary.Text.Replace("597", "a");
81     }
82
83     if (a.Contains("598"))
84     {
85         sonmesajBinary.Text = sonmesajBinary.Text.Replace("598", "b");
86     }
87
88     if (a.Contains("599"))
89     {
90         sonmesajBinary.Text = sonmesajBinary.Text.Replace("599", "c");
91     }
92
93     if (a.Contains("600"))
94     {
95         sonmesajBinary.Text = sonmesajBinary.Text.Replace("600", "d");
96     }
97
98     if (a.Contains("601"))
99     {
100        sonmesajBinary.Text = sonmesajBinary.Text.Replace("601", "e");
101    }
102
103    if (a.Contains("602"))
104    {
105        sonmesajBinary.Text = sonmesajBinary.Text.Replace("602", "f");
106    }
107
108    if (a.Contains("603"))
109    {
110        sonmesajBinary.Text = sonmesajBinary.Text.Replace("603", "g");
111    }
112

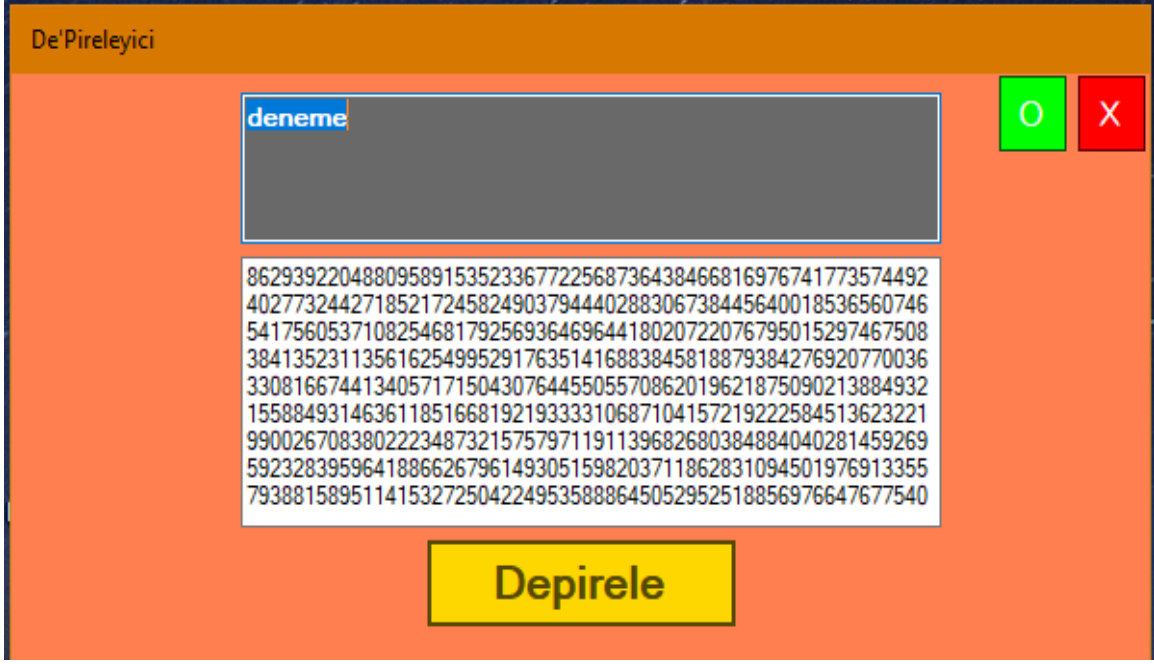
```

Burada, verinin sayı karşılıklarının karakteri a adlı değişkene atanır ve “If Contains(eğer içeriyor ise)” tekniği kullanılarak sayılar, atanan karakter değerlerine dönüştürülüp sonmesajBinary adlı textBox’un içerisinde depolanmaktadır.

### 3.5. Proje Ekran Görüntüleri



**Görsel 1. Pireleyici**



**Görsel 2. De'Pireleyici**

#### 4. Sonuçlar ve Tartışma

Proje, hızlı ve uyumlu bir şekilde çalışabilmesi için c#, Windows platformu ile kullanıldı. Sistem, demo amaçlı Pi'releyici ve DePi'releyici bir arada hazırlandı ancak bu sistemler ayrı ayrı cihazlarda şifre ile erişilebilen sistemler olacak ve bu sayede güvenlik yönünden bir takım açıklıklar giderilmiş olacaktır. Hardcoded olduğundan bazen geri dönüştürme aşamasında belli karakterler tekrar edilince bunların yanlış sıra ile dönüştüğü tespit edilmiş ve tüm karakterlerin sayı karşılıkları int array içine alınarak teker teker dönüştürülmüş, sorun düzeltilmiştir.

#### 5.Öneriler

Projenin, manuel olarak bir dönüştürme uygulaması olmasındansa, veri girildikçe arkaplanda otomatik olarak dönüştürülüp gönderilmeli ve yine otomatik olarak pi kodu algılandığında otomatik geri dönüştürülmelidir. Proje, geliştirmeye açıktır.

Proje, başka sistemlere entegre edilerek, o sistemin bir alt sistemi haline gelebilir.

Proje, yalnızca string(metin) türündeki verileri değil de her çeşit veriyi okuyup dönüştürebilir hale getirilebilir. Bu kullanım alanını oldukça geliştirecektir.(Örn: Ses, Görüntü...)

## Kaynakça

<https://ssd.eff.org/tr/module/%C5%9Fifreleme-nedir> erişim tarihi: 08.10.2018

<https://docs.microsoft.com/en-us/visualstudio/releases/notes/vs2017-relnotes/> erişim tarihi: 28.11.2018

<https://www.digitalunite.com/technology-guides/computer-basics/using-computer/what-windows> erişim tarihi: 28.11.2018

Atalay Keleştemur C# Dilinin Tarihçesi (21.05.2016) adlı çevrimiçi makalesi  
<https://gelecegiyazanlar.turkcell.com.tr/konu/c-sharp/egitim/c-101/c-dilinin-tarihcesi>  
erişim tarihi: 02.10.2018

<http://www.csharpnedir.com/> erişim tarihi: 05.11.2018

<https://www.piday.org/million/> erişim tarihi: 09.10.2018

<https://tr.khanacademy.org/computing/computer-science/cryptography/ciphers/a/ciphers-vs-codes> erişim tarihi: 02.10.2018

<https://searchsecurity.techtarget.com/definition/encryption> erişim tarihi: 05.10.2018