

# LZ77 ve DEFLATE Sıkıştırma Algoritması

Furkan Aydoğan  
Bilgisayar Mühendisliği, Kocaeli Üniversitesi  
180202085  
[180202085@kocaeli.edu.tr](mailto:180202085@kocaeli.edu.tr)

Sefa Berke Kara  
Bilgisayar Mühendisliği, Kocaeli Üniversitesi  
180202086  
[180202086@kocaeli.edu.tr](mailto:180202086@kocaeli.edu.tr)

## Giriş

LZ77, DEFLATE sıkıştırma algoritmaları uygulaması “metin.txt” sıkıştırılmamış dosyasını okur. Bu dosyanın içindeki satır bilgilerine program tarafından erişim sağlanır. Dosyalardaki her bir satırda bulunan bilgiler belleğe kayıt edilir. Dosya LZ77 ve Deflate algoritmaları ile sıkıştırılır. LZ77 algoritması tarafından sıkıştırılmış dosyanın çıktıları “encode.txt” dosyasına Deflate tarafından sıkıştırılmış dosyanın çıktısı ise “cikti.txt” dosyasına yazdırılır.

## Özet

LZ77, DEFLATE sıkıştırma algoritmaları uygulaması Okuma tamamlandıktan sonra proje tarafından istenilen bölüme geçiş yapılır. İlk olarak LZ77 algoritması metin belgesinde tekrar eden bölümleri ortadan kaldırarak buna yönelik sıkıştırılmış metin belgesini yazdırır. İkinci olarak ise Deflate algoritması LZ77+Huffman algoritmasını kullanarak metin belgesini sıkıştırır “cikti.txt” dosyasında da yazılı olarak bastırır.

## Temel Bilgiler

Windows 10 işletim sistemine sahip bilgisayar ile geliştirme yapılmıştır. Program C programlama dilinde geliştirilmiş olup, tümleşik geliştirme ortamı olarak “CodeBlocks” kullanılmıştır

## Tasarım

LZ77, DEFLATE sıkıştırma algoritmaları uygulaması programlanma aşamaları altta belirtilen başlıklar altında açıklanmıştır

## Algoritma

LZ77 ve DEFLATE olmak üzere 2 programdan oluşur .

LZ77 programı algoritması için “metin.txt” dosyası içindeki verileri tek tek okur. Okunan karakterler bir diziye atanır. Struct tokens tipindeki “kodla” fonksiyonuna parametre olarak gönderilen “metin” dizisi kaç karakterden oluştuğunu tutan “metin\_boyutu” ve token sayısı ile LZ77 algoritmasındaki sıkıştırma aşamasına geçilir. İlk olarak token için boyut bellekte yer ataması yapılır ve maximum metin sayısı belirlenir. Fonksiyon for döngüsü içinde başlangıç karakterinden son karaktere kadar döner.

LZ77 algoritmasında tekrar eden metinleri bulabilmek için ileri\_tampon(look ahead buffer) ve arama\_tampon(search buffer) adlı 2 tane “pointer char” tipinde

değişkenlere ihtiyaç duyulur. Program başlamadan “OFFSET=31” ve “LENGTH=7” olmak üzere tanımlanan değişkenler LZ77 algoritması içindeki for döngüsünde İleri tampon ‘un döngünün içinde dönerken o anki adres değerinden “OFFSET” çıkartılır ve arama tamponu değişkenine eşitlenir. Eğer “metin” adresi “arama\_tampon” adresinden daha küçük ise “arama\_tampon” değişkeni “metin” değişkenine eşitlenir. Bunun yapılmasının nedeni “arama\_tampon”un metnin dışına çıkmasına engel olmaktır. Ardından en fazla eşleşmenin olabileceği adresi tutan “max\_eslesme” değişkeni “ileri\_tampon”a eşitlenir. Bu işlemler algoritma tarafından tamamlandıktan sonra program “arama\_tampon (serach\_buffer)”, “ileri\_tampon(look\_ahaed buffer)”a kadar while döngüsünde döndürülerek “arama\_tampon” içinde arama yapılır. Döngü tamamlandıktan sonra eğer eşleşmenin içine metnin son karakteri de dahil olmuşsa, tokenin içine bir karakter koyabilmek için eşleşmesi kısaltmamız gerekiyor. Bunun içinde “metin” ile “sınır” değişkenleri toplanır bu toplanan değişkenlerden “ileri\_tampon” ve “-1” çıkartma yapıp “max\_lenght” yani maximum uzunluğa eşitlenir. Bu işlemlerde tamamlandıktan sonra eğer gerekiyorsa hafızada yer açılır. Bulunan eşleşmeye göre token oluşturur ve oluşturulan “token” diziye kaydedilir. Fonksiyon en son olarak da “token” return edilir. “encode.txt” dosyasına LZ77 algoritması ile sıkıştırılan metinler bastırılır. Bundan sonra da “binary.txt” adlı dosya oluşturularak sıkıştırılmış dosyanın binary hali bastırılır ve konsol ekranında sıkıştırılmış hali kullanıcıya belirtilir.

DEFLATE programı LZSS programı algoritması için “metin.txt” dosyası içindeki verileri tek tek okur. Okunan karakterler bir diziye atanır. Struct tokens tipindeki “kodla” fonksiyonuna parametre olarak gönderilen “metin” dizisi kaç karakterden oluştuğunu tutan “metin\_boyutu” ve token sayısı ile LZSS algoritmasındaki sıkıştırma aşamasına geçilir. Sıkıştırma tamamlandıktan sonra “token” struct değişkeni return edilir. Sıkıştırılmış metin “encode.txt”ye bastırılır. Eğer karakter sayısı 3 den daha küçük ise karakterler olduğu gibi bastırılır. Üçten daha yüksek ise sıkıştırılmış hali bastırılır. LZSS ile sıkıştırılmış “encode.txt” dosyasının her karakteri okunur ve belleğe kayıt edilir. Karakterin ASCII karşılığı 144 ‘den daha küçük ise if bloğuna girer. Eğer if bloğuna giren karakterin ASCII karşılığı 48 ile

57 arasında ise yani bir sayı ise gerekli if bloğuna girer. Burada ilk sayı bir değişkende tutulur. Tutulan bu ilk sayı mesafeyi temsil eder. Bu sayı dist() adlı fonksiyona gönderilir. dist() fonksiyonunda static DEFLATE mesafe tablosuna göre code değerleri bulunmaktadır. Gönderilen sayının code değeri return edilerek değişkende tutulur. Ardından bu tutulan değişken “binary()” fonksiyonuna gönderilerek DEFLATE code’nun 5 basamaklı binary karşılığı “cikti.txt” dosyasına yazdırılır. Tutulan aynı sayı bu seferde “extra()” fonksiyonuna yollanır. “extra()” fonksiyonunda DEFLATE mesafe tablosuna göre extra bitler tanımlıdır. Gönderilen sayıya göre extra bit tutularak “binary()” fonksiyonuna gönderilir ve tekrardan binary karşılığı dosyaya yazdırılır. Ve if koşulundan çıkılır. Ardından bir sonraki sayı yeniden “sayı” değişkeninde tutulur ve else bloğuna girilir. Burada DEFLATE uzunluk tablosundaki kod karşılığına göre if bloğunda kontrol edilir. Eğer uzunluğun kod karşılığı 257 ile 280 arasındaysa if bloğuna girer. Uzunluğun kod karşılığı 257 ile 280 arasında hangi sayı ise o sayının binary karşılığı 7 bit olacak şekilde “cikti.txt” dosyasına yazdırılır. Eğer 280’den daha büyük ise o kodun karşılığı 8 bit o olacak şekilde yazdırılır. Bu bloklar tamamlandıktan sonra eğer ASCII karşılığı 48 ile 57 arasında değil ise else bloğuna girer. Mod 2’deki Huffman Kodlarındaki Edoc 0’a uyarlayabilmek için tutulan char karakterinin ASCII karşılığına 48 eklenerek “temp” değişkenine atanır. Bu atanan değer binary olarak Edoc’un bit uzunluğuna göre dosyaya yazdırılır. Eğer sayının “edoc” karşılığı 144 ile 255 arasında ise sayının ASCII değerine 400 eklenir ve binary fonksiyonuna gönderilerek dosyaya edoc un bit karşılığı şeklinde yazdırılır. Dosya sonunun bir karşılığı en son eklenir.

## Sözde Kod (LZ77)

1. LZ77 programı algoritması için “metin.txt” dosyası içindeki verileri tek tek oku.
2. . Okunan karakterler bir diziye atanır. “struct tokens” tipindeki “kodla” fonksiyonuna parametre olarak gönder.
3. Gönderilen “metin” dizisi kaç karakterden oluştuğunu tutan “metin\_boyutu” ve token sayısı ile LZ77 algoritmasındaki sıkıştırma aşamasına geç.

4. token için boyut bellekte yer ataması yapılır ve maximum metin sayısı belirlir.
5. Fonksiyon for döngüsü içinde başlangıç karakterinden son karaktere kadar döndür.
6. LZ77 algoritmasında tekrar eden metinleri bulabilmek için ileri\_tampon(look ahead buffer) ve arama\_tampon(search buffer) adlı 2 tane "struct tokens" tipinde tanımla.
7. Program başlamadan "OFFSET =31" ve "LENGHT =7" olmak üzere tanımlanan değişkenler LZ77 algoritması içindeki for döngüsünde İleri tampon 'un döngünün içinde dönerken o anki adres değerinden "OFFSET" çıkart.
8. Arama tamponu değişkenine eşitle.
9. Eğer "metin" adresi "arama\_tampon" adresinden daha küçük ise "arama\_tampon" değişkeni "metin" değişkenine eşitle
10. işlemler algoritma tarafından tamamlandıktan sonra program "arama\_tampon (search\_buffer)" , "ileri\_tampon(look ahead buffer)"a kadar while döngüsünde döndürülerek "arama\_tampon" içinde arama yap.
11. "metin" ile "sınır" değişkenleri toplanır bu toplanan değişkenlerden "ileri\_tampon" ve "-1" çıkartma yap ve "max\_lenght" yani maximum uzunluğa eşitle.
12. Bu işlemlerde tamamlandıktan sonra eğer gerekiyorsa hafızada yer aç.
13. Bulunan eşleşmeye göre token oluştur ve oluşturulan "token" diziyeye kaydet.
14. . Fonksiyon en son olarak da "token" return et.
15. "encode.txt" dosyasına LZ77 algoritması ile sıkıştırılan metinler bastır.
16. . Bundan sonra da "binary.txt" adlı dosya oluşturularak sıkıştırılmış dosyanın binary halini bastır.
17. Konsol ekranında sıkıştırılmış hali kullanıcıya göster

. Sözde Kod (DEFLATE)

1. DEFLATE programı LZSS programı algoritması için "metin.txt" dosyası içindeki verileri tek tek oku
2. Okunan karakterler bir diziye ata
3. . Struct tokens tipindeki "kodla" fonksiyonuna parametre olarak gönderilen "metin" dizisi kaç karakterden oluştuğunu tutan "metin\_boyutu" ve token sayısı ile LZSS algoritmasındaki sıkıştırma aşamasına geç
4. Sıkıştırma tamamlandıktan sonra "token" struct değişkenini return et.
5. Sıkıştırılmış metin "encode.txt"ye bastır.
6. Eğer karakter sayısı 3 den daha küçük ise karakterler olduğu gibi bastır.
7. Üçten daha yüksek ise sıkıştırılmış hali bastırılır. LZSS ile sıkıştırılmış "encode.txt" dosyasının her karakteri oku ve belleğe kaydet.
8. Karakterin ASCII karşılığı 144 'den daha küçük ise if bloğuna gir.
9. Eğer if bloğuna giren karakterin ASCII karşılığı 48 ile 57 arasında ise if bloğuna gir
10. Burada ilk sayıyı bir değişkende tut.
11. Bu tutulan değişken "binary()" fonksiyonuna gönder DEFLATE code'nun 5 basamaklı binary karşılığı "cikti.txt" dosyasına yazdır.
12. Tutulan aynı sayı bu seferde "extra()" fonksiyonuna yolla.
13. "extra()" fonksiyonunda DEFLATE mesafe tablosuna göre extra bitler tanımla.
14. Gönderilen sayıya göre extra bit tutularak "binary()" fonksiyonuna gönder ve tekrardan binary karşılığı dosyaya yazdır.
15. Bir sonraki sayı yeniden "sayı" değişkeninde tut ve else bloğuna gir.
16. DEFLATE uzunluk tablosundaki kod karşılığına göre if bloğunda kontrol et.
17. Eğer uzunluğun kod karşılığı 257 ile 280 arasındaysa if bloğuna gir.
18. . Uzunluğun kod karşılığı 257 ile 280 arasında hangi sayı ise o sayının binary karşılığı 7 bit olacak şekilde "cikti.txt" dosyasına yazdır.
19. .eğer ASCII karşılığı 48 ile 57 arasında değil ise else bloğuna gir.
20. Mod 2'deki Huffman Kodlarındaki Edoc 0'a uyarlayabilmek için tutulan char karakterinin ASCII karşılığına 48 eklenerek temp değişkenine ata.

21. atanır. Bu atanan değer binary olarak Edoc'un bit uzunluğuna göre dosyaya yazdır.

22. . Eğer sayının edoc karşılığı 144 ile 255 arasında ise sayının ASCII değerine 400 eklenir ve binary fonksiyonuna gönderilerek dosyaya edoc un bit karşılığı şeklinde yazdır.

23. Dosya sonunun bir karşılığı en son ekle.

Daha önce sözel olarak öğrendiğimiz sıkıştırma algoritmalarını bu proje sayesinde pratiğe dökme şansı bulduk.

Bu ve buna benzer algoritmalar bir daha karşımıza çıktığında bu proje sayesinde daha hızlı düşünüp ne gibi yol izleyeceğimizi kısa sürede düşünmemizde yardımcı olacaktır.

## Kullanılan Fonksiyonlar (DEFLATE)

### Karşılaştığımız sorunlar:

LZ77 algoritması için kayan pencere şeklinde arama nasıl yapacağımız konusunda bir takım sorunlar yaşadık.

LZ77 için ilk aşamada pencere boyutunu kaç karakter geriye götürüp kaç karakter eşleşme olacağı konusunda fikir sahibi olamadık.

Araştırmalarımız sonucunda 32 karakter geriye götürüp eşleşmeyi de en fazla 8 karakter olacak şekilde ayarladık.

DEFLATE algoritması için internette algoritmayı anlayabileceğimiz kadar kaynak bulamadık bu algoritma için uzun bir süre araştırmalar yaptık. Bunun sonucunda belli bir fikir sahibi olarak hazır kodlanmış static huffman kodu ile (MOD 01) yapmaya karar verdik.

### Projenin Bize Kattığı Yararlar:

Lz77 ve Deflate algoritmaları bize bit bazında işlemler yapmamızda katkı sağladı.

Daha önce sözel olarak öğrendiğimiz sıkıştırma algoritmalarını bu proje sayesinde pratiğe dökme şansı bulduk.

Kodla():

LZSS algoritmasını barındırır.

Binary():

Alınan değeri binary haline çevirir.

Benzerlik():

DEFLATE uzunluk tablosundaki kodları ve extra bitleri barındırır.

Dist():

DEFLATE mesafe tablosundaki 5 bitlik kodları barındırır.

Extra():

DEFLATE mesafe tablosundaki extra bitleri barındırır.

## Kullanılan fonksiyonlar (LZ77)

Kodla()

LZ77 Sıkıştırma Algoritması

LZ77 algoritmasını barındırır.

<https://ysar.net/algoritma/lz77.html>

## Sonuçlar

LZ77 Compression Algorithm

LZ77 algoritması “metin.txt” dosyasını okur ve “encode.txt” de tokenleri bastırır ve “binary.txt”de binary halini bastırır.

[https://docs.microsoft.com/en-us/openspecs/windows\\_protocols/ms-wusp/fb98aa28-5cd7-407f-8869-a6cef1ff1ccb](https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wusp/fb98aa28-5cd7-407f-8869-a6cef1ff1ccb)

Deflate Algoritması “metin.txt” dosyasından okur ve LZSS algoritmasının sıkıştırılmış halini “encode.txt” dosyasına yazdırır. Static Deflate Tablolarını kullanarak da “cikti.txt”ye binary şeklinde yazdırır.

puff.c By Mark Adler

<https://github.com/madler/zlib/blob/master/contrib/puff/puff.c>

## Kaynakça

DEFLATE Encoding with static Huffman Codes

<https://stackoverflow.com/questions/17398931/deflate-encoding-with-static-huffman-codes?rq=1>

DEFLATE Compressed Data Format Specification

<https://www.ietf.org/rfc/rfc1951.txt>

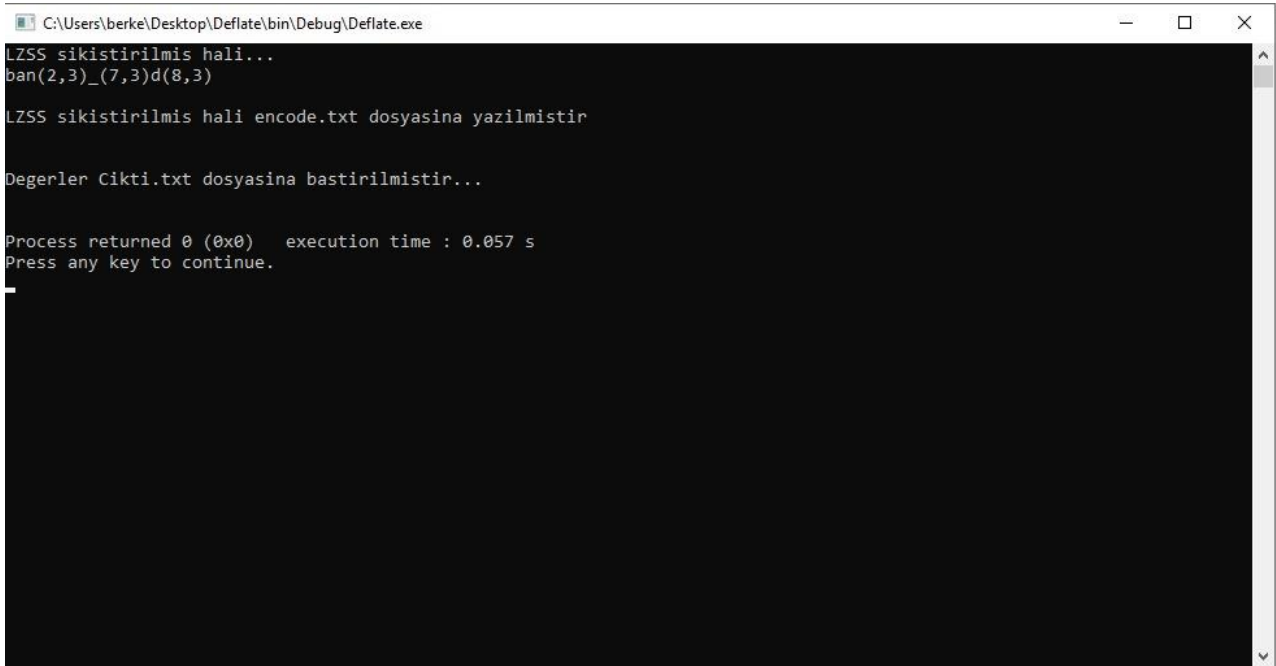
Handbook of Data Compression

<https://ceng2.ktu.edu.tr/~cakir/files/sistemlab/Handbook%20of%20Data%20Compression,%205th%20Edition.pdf>

DEFLATE Compression Algorithm

<http://www.integpg.com/deflate-compression-algorithm/>

## Ekran Görüntüleri

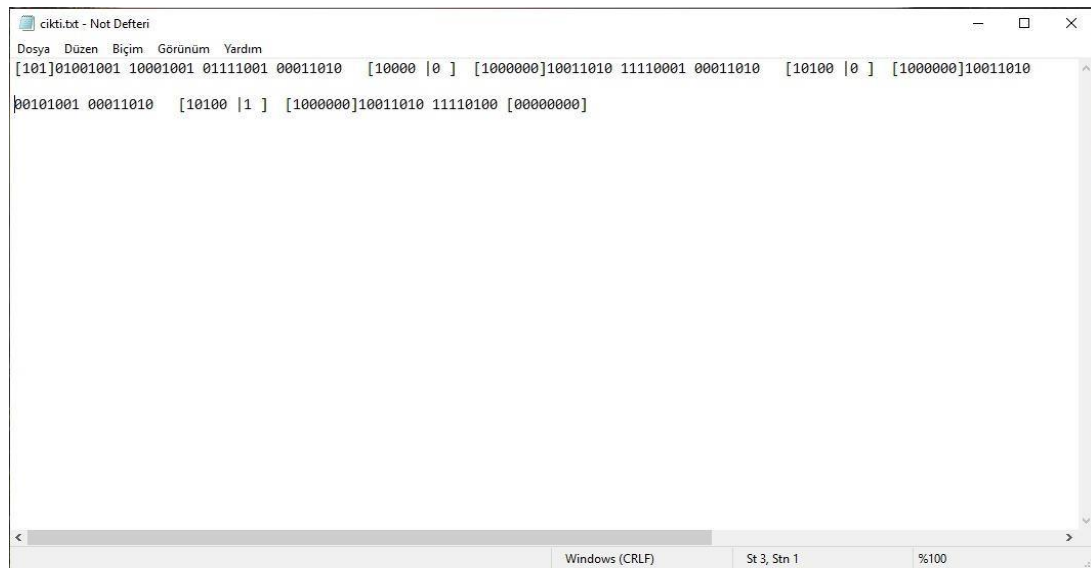


```
C:\Users\berke\Desktop\Deflate\bin\Debug\Deflate.exe
LZSS sıkıştırılmış hali...
ban(2,3)_(7,3)d(8,3)

LZSS sıkıştırılmış hali encode.txt dosyasına yazılmıştır

Değerler Cikti.txt dosyasına bastırılmıştır...

Process returned 0 (0x0)   execution time : 0.057 s
Press any key to continue.
```



```
cikti.txt - Not Defteri
Dosya Düzen Biçim Görünüm Yardım
[101]01001001 10001001 01111001 00011010 [10000 0 ] [1000000]10011010 11110001 00011010 [10100 0 ] [1000000]10011010
00101001 00011010 [10100 1 ] [1000000]10011010 11110100 [00000000]

Windows (CRLF) St 3, Stn 1 %100
```

- İlk Köşeli parantez olan [101] Header'dır. Header'dan sonraki ilk köşeli parantez mesafeyi(distance), ikinci köşeli parantez ise uzunluğu(lenght)temsil eder. RFC 1951 3.1.1'deki açıklamaya göre bütün karakterlerin binary karşılığının ters çevrilmiş hali bastırılmıştır. Sondaki köşeli parantez olan [00000000] ise End Code 'dur.

## LZ77

```
binary_encode.txt - Not Deferi
Dosya Düzen Biçim Görünüm Yardım
a!d b r c d a

Windows (CRLF) St 1, Stn 1 %100

encode.txt - Not Deferi
Dosya Düzen Biçim Görünüm Yardım
0,0,a 0,0,b 0,0,r 3,1,c 5,1,d 7,4,a

Windows (CRLF) St 1, Stn 1 %100

C:\Users\berke\Desktop\Prolab2_Z\main.exe
LZ77 ile sıkıştırılmış hali
<0,0,a>
<0,0,b>
<0,0,r>
<3,1,c>
<5,1,d>
<7,4,a>

sıkıştırma oranı =%55

Process returned 0 (0x0) execution time : 0.143 s
Press any key to continue.
```