

# VERİ DEPOLAMA VE SIKIŞTIRMA

Sunumun amacı; bilgisayarların insan algısını (görme, duyma, okuma) nasıl 0 ve 1'lere dönüştürdüğünü ve bu verileri nasıl "paketlediğini" anlamaktır.

# VERİ NEDİR?

Veri, bilgisayarda saklanan ve işlenen ham bilgilerdir; tek başına anlamı yoktur, işlendiğinde bilgiye dönüşür.



# VERİ NASIL DEPOLANIR?

Veri, bilgisayarda dosyalar, veritabanları ve depolama ortamları (HDD, SSD, bulut) üzerinde ikili (0 ve 1) biçimde depolanır.

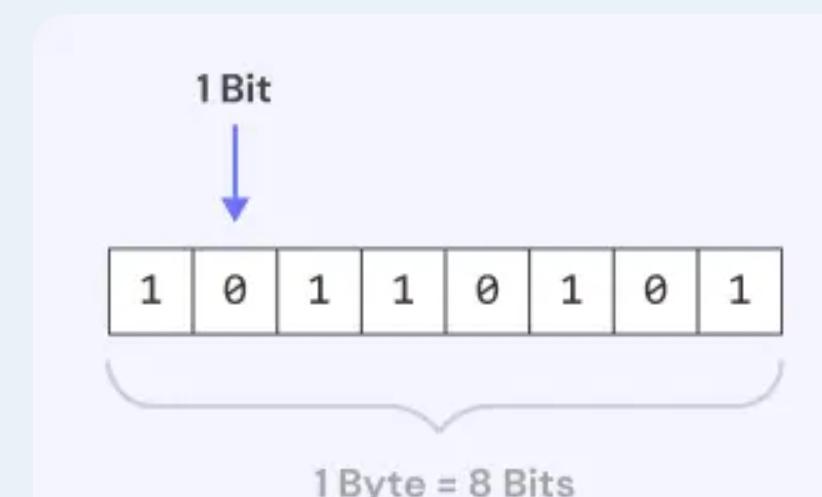


# **VERİ NEDEN DEPOLANIR?**

Veri, ileride tekrar kullanılmak, analiz yapmak, karşılaştırma yapmak ve karar verebilmek için depolanır. Depolanmayan veri kaybolur ve sistem için degersizdir.

# BİTLER VE BİT DESENLERİ

Bit, bilgisayardaki en küçük veri birimidir ve 0 veya 1 değerini alır. Bit desenleri, bu 0 ve 1'lerin belirli bir sırayla dizilmesiyle verinin (sayı, harf, resim vb.) temsil edilmesini sağlar.



# İKİLİK SİSTEM

İkilik sistem, yalnızca 0 ve 1 kullanılarak sayıların ve verinin temsil edildiği sayı sistemidir; bilgisayarlar veriyi bu sistemle işler ve depolar.



# HEXADECIMAL (ON ALTLIK) SİSTEM

Hexadecimal (on altılık) sistem, 0-9 ve A-F sembollerini kullanarak ikilik sayıların daha kısa ve okunabilir şekilde gösterilmesini sağlar.

Decimal	4-Bit Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

# METİNLERİN DEPOLANMASI

Metinler, bilgisayarda her karakterin bir sayı ile temsil edilmesi (örneğin ASCII veya Unicode) ve bu sayıların ikilik sistemde saklanmasıyla depolanır.

Decimal	4-Bit Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

# ASCII NEDİR?

- ASCII (American Standard Code for Information Interchange), 1960'lı yıllarda ABD'de geliştirilen bir karakter kodlama standardıdır.
- Toplam 7 bit kullanır ve 128 karakteri destekler.
- Bu karakterler; İngiliz alfabesi harfleri (A-Z, a-z), rakamlar (0-9), noktalama işaretleri ve kontrol karakterlerinden oluşur.
- ASCII Türkçe karakterleri desteklemez; bu yüzden daha sonra Unicode geliştirilmiştir.

**ASCII TABLE**

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

# UNICODE NEDİR?

- Unicode, ASCII'nin yetersiz kalması nedeniyle geliştirilen, dünyadaki tüm dilleri ve sembollerini kapsaması amaçlanan evrensel bir karakter kodlama standardıdır.
- 1991'de yayınlanmıştır ve sabit bir bit sayısına bağlı değildir; UTF-8, UTF-16 ve UTF-32 gibi farklı kodlama biçimleri kullanır.
- Türkçe karakterler, matematiksel semboller ve emojiler dahil çok geniş bir karakter kümesini destekler.

# DOSYA FORMATLARI VE UZANTILAR

- Dosya formatları, verinin dosya içinde nasıl düzenlenip saklandığını tanımlar;
- dosya uzantıları ise bu formatın hangi tür veri içerdigini ve hangi programlarla açılacağını belirtir.
- Örnek:

*txt → düz metin*

*jpg → resim*

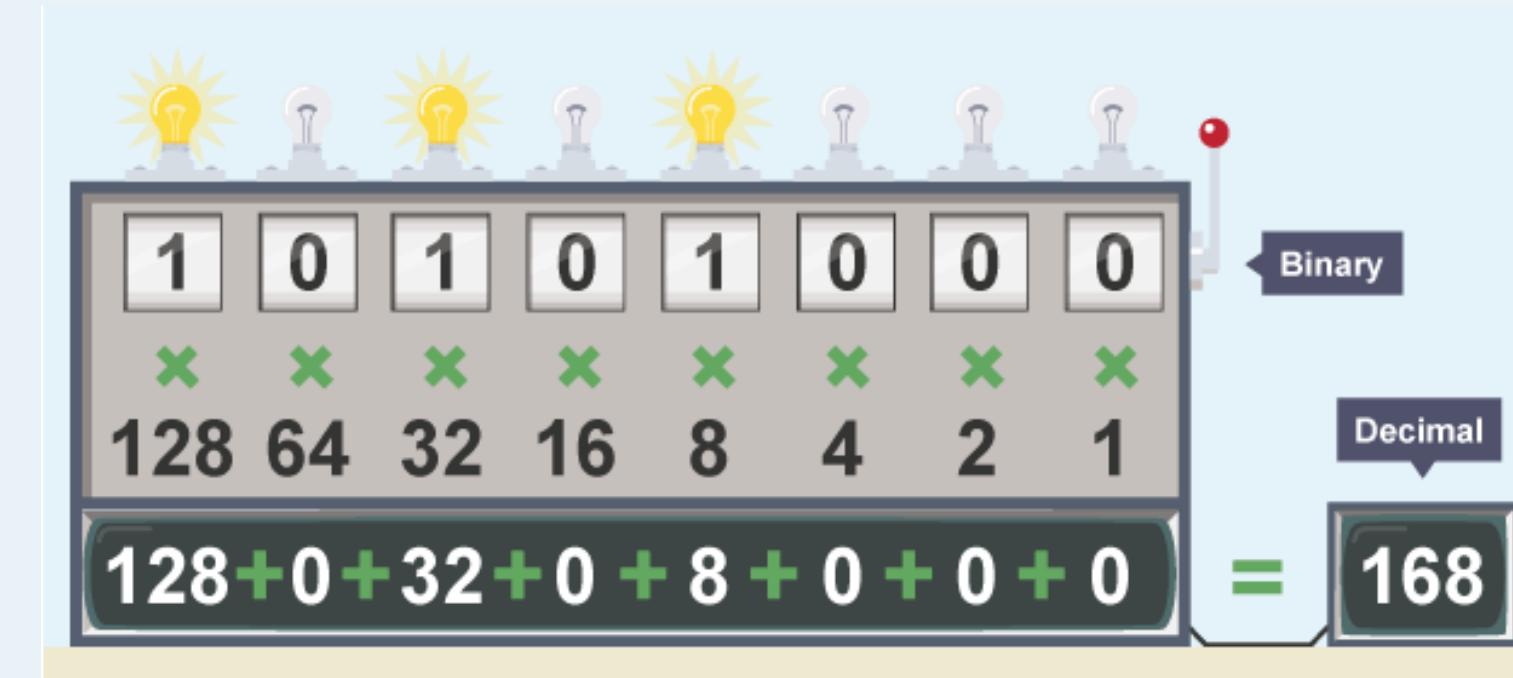
*mp3 → ses*



- Uzanti sadece bir etikettir; asıl belirleyici olan dosyanın iç yapısıdır.

# SAYISAL VERİLERİN DEPOLANMASI

Sayısal veriler, bilgisayarda ikilik sistem kullanılarak depolanır; tam sayılar ve ondalıklı sayılar için farklı temsil yöntemleri (ör. iki'nin tümleyeni, kayan noktalı gösterim) kullanılır.



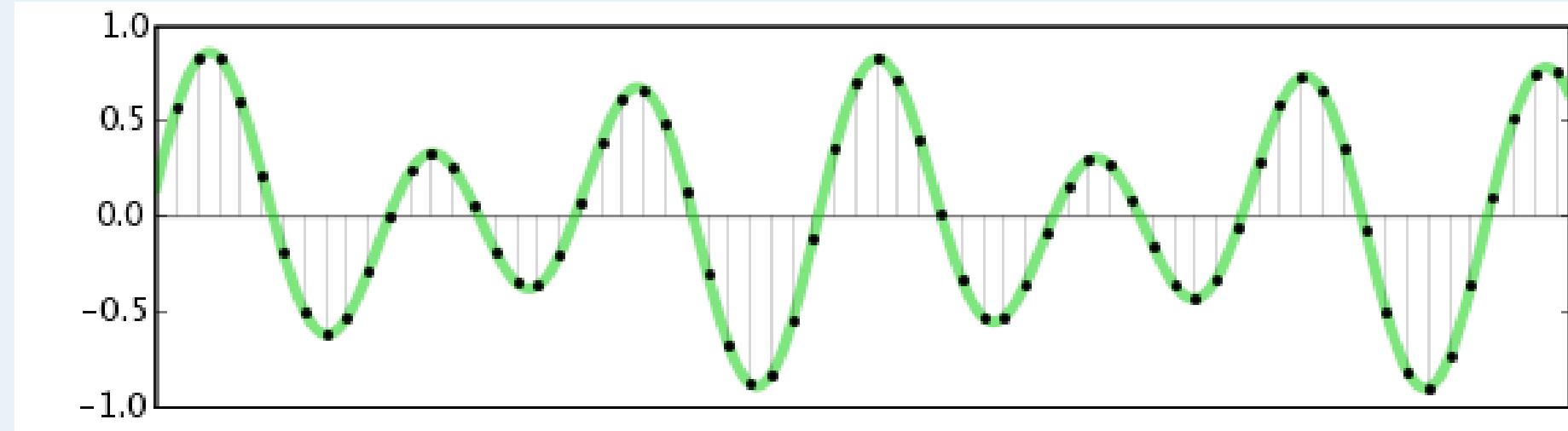
# RESİMLERİN DEPOLANMASI

- Resimler, piksellerden oluşur ve her pikselin renk bilgisi sayısal değerler olarak ikilik sistemde depolanır.
- Depolama biçimi; çözünürlük, renk derinliği ve kullanılan dosya formatına (PNG, JPG vb.) bağlıdır.
- Yani resimler bir görüntü değil, sayı tablolarıdır.

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	25
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	249	255	240	255	129	0	5	0
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	0	14	1	0	6	6	0

# SESLERİN DEPOLANMASI

- Sesler, analog dalgaların sayısal örneklenmesi ile elde edilir ve bu örnekler ikilik sistemde sayısal değerler olarak depolanır.
- Ses kalitesi; örnekleme hızı, bit derinliği ve kullanılan dosya formatına (WAV, MP3 vb.) bağlıdır.



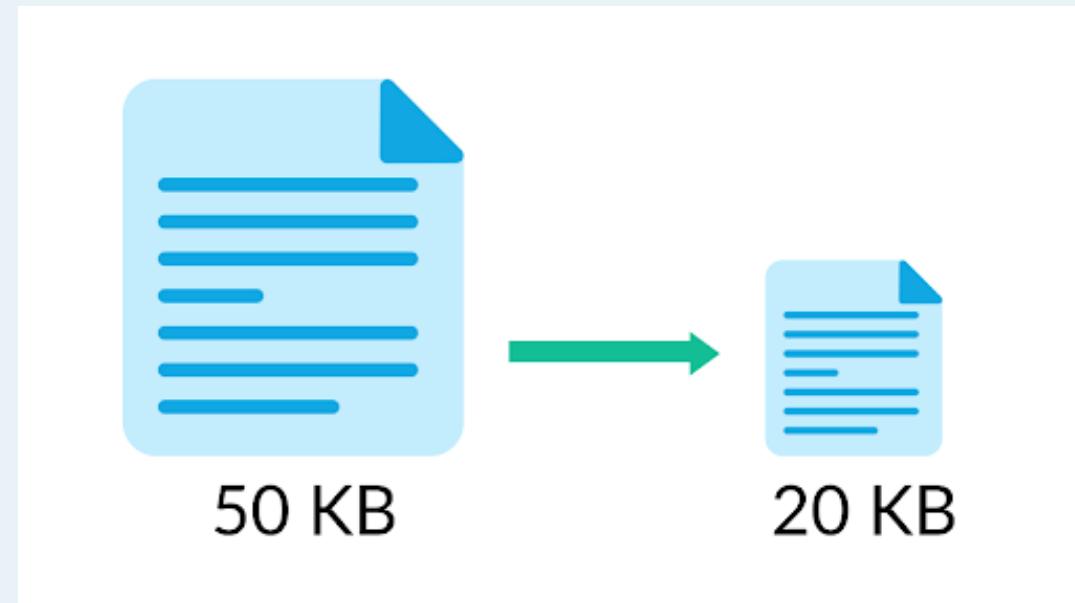
# VERİ KAPASİTESİ VE BİRİMLERİ

- Veri kapasitesi, depolama ortamının saklayabileceği veri miktarını ifade eder ve bit temel alınarak ölçülür.
- Bit, Byte (8 bit), KB, MB, GB ve TB gibi birimler, kapasitenin büyüklüğünü göstermek için kullanılır.

Data Unit	Size (Bytes)	Size (Approx.)
Bit (b)	1	-
Byte (B)	8 bits	-
Kilobyte (KB)	1,000 bytes	$10^3$ bytes
Megabyte (MB)	1,000,000 bytes	$10^6$ bytes
Gigabyte (GB)	1,000,000,000 bytes	$10^9$ bytes
Terabyte (TB)	1,000,000,000,000 bytes	$10^{12}$ bytes
Petabyte (PB)	1,000,000,000,000,000 bytes	$10^{15}$ bytes
Exabyte (EB)	1,000,000,000,000,000,000 bytes	$10^{18}$ bytes
Zettabyte (ZB)	1,000,000,000,000,000,000,000 bytes	$10^{21}$ bytes
Yottabyte (YB)	1,000,000,000,000,000,000,000,000 bytes	$10^{24}$ bytes

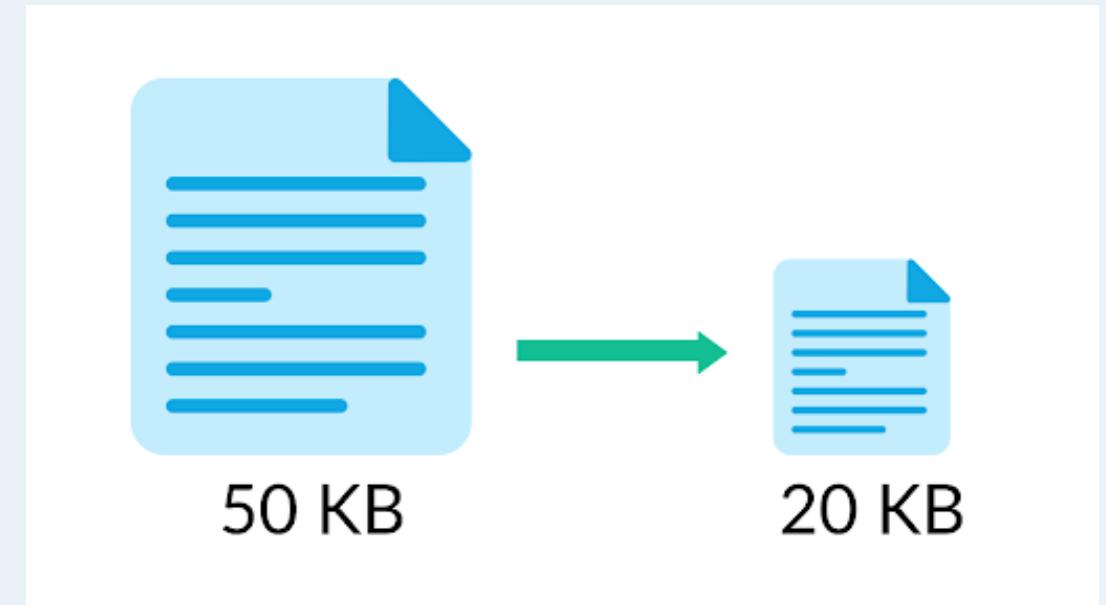
# VERİ NEDEN SIKIŞTIRILIR?

- Veri, daha az yer kapaması, daha hızlı aktarılması ve depolama maliyetini düşürmek için sıkıştırılır.



# VERİ NASIL SIKIŞTIRILIR?

- Veri sıkıştırma, tekrar eden veya gereksiz bilgilerin daha az bit kullanacak şekilde kodlanmasıyla yapılır.
- Bu yöntem ikiye ayrılır:
- Kayıpsız sıkıştırma: Veri açıldığında ilk hâliyle birebir geri elde edilir. (ZIP, PNG)
- Kayıplı sıkıştırma: İnsan algısının fark etmeyeceği bilgiler bilerek atılır, daha yüksek sıkıştırma sağlanır. (JPEG, MP3)



# SIKİŞTIRMA ORANI

- Sıkıştırma oranı, sıkıştırma işleminden sonra verinin ne kadar küçüldüğünü gösteren ölçütür ve genellikle orijinal boyut / sıkıştırılmış boyut şeklinde ifade edilir.

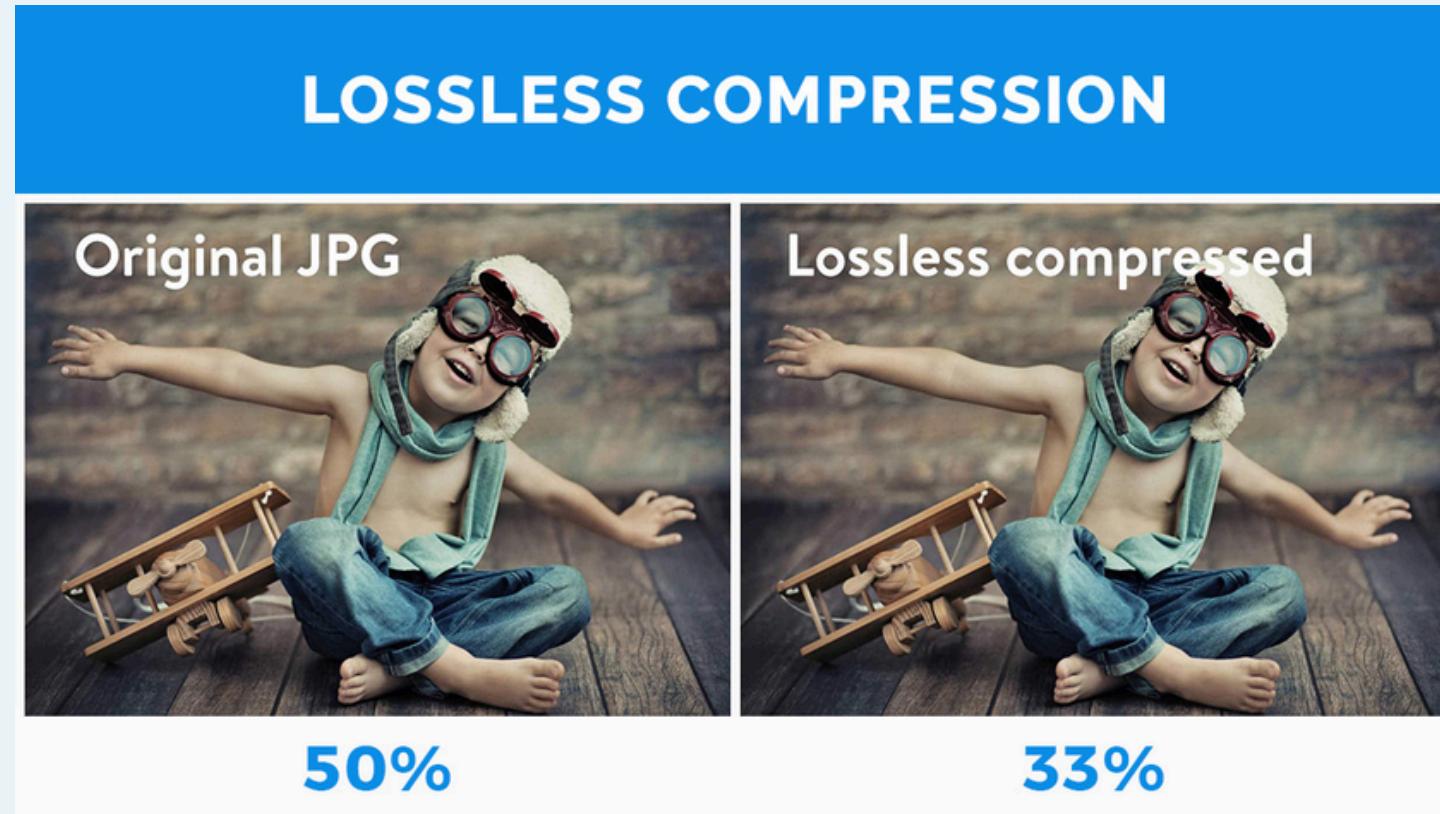
# KAYIPLI SIKIŞTIRMA

- Kayıplı sıkıştırma, verinin insan algısının fark etmesi zor olan kısımlarını kalıcı olarak silerek dosya boyutunu küçültme yöntemidir.
- Geri açıldığında veri orijinaliyle birebir aynı değildir; daha küçük ama daha düşük kalitelidir.



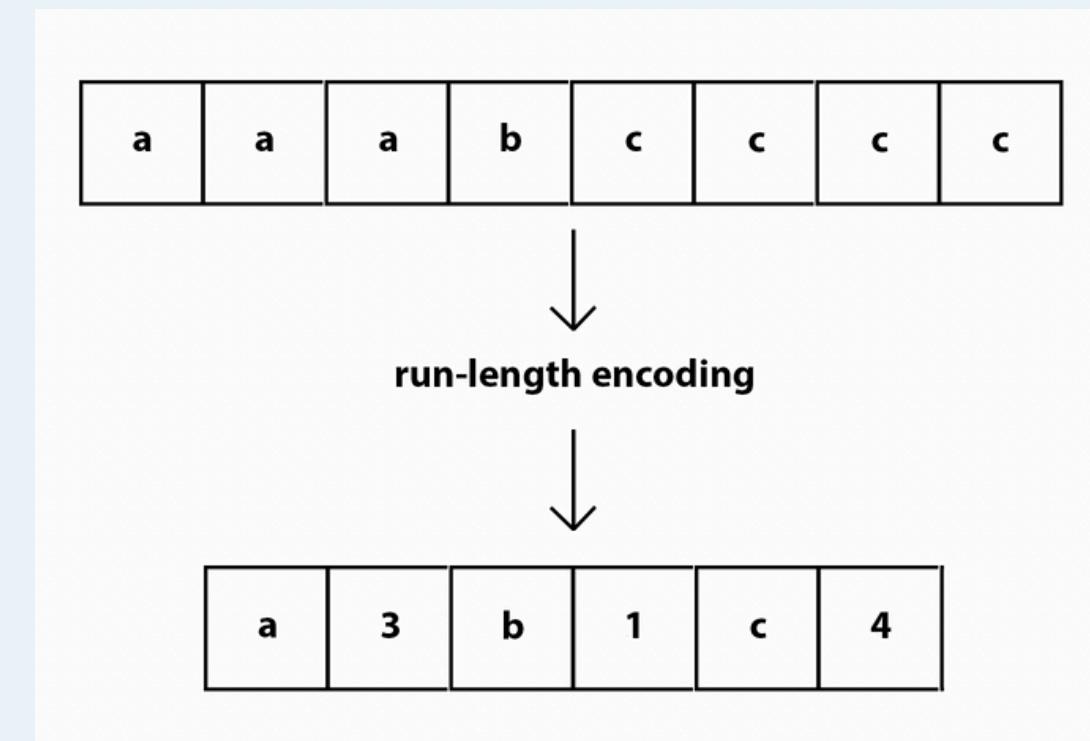
# KAYIPSIZ SIKIŞTIRMA

- Kayıpsız sıkıştırma, verinin hiçbir bilgi kaybı olmadan daha az yer kaplayacak şekilde sıkıştırılmasıdır.
- Açıldığında veri orijinaliyle birebir aynıdır.



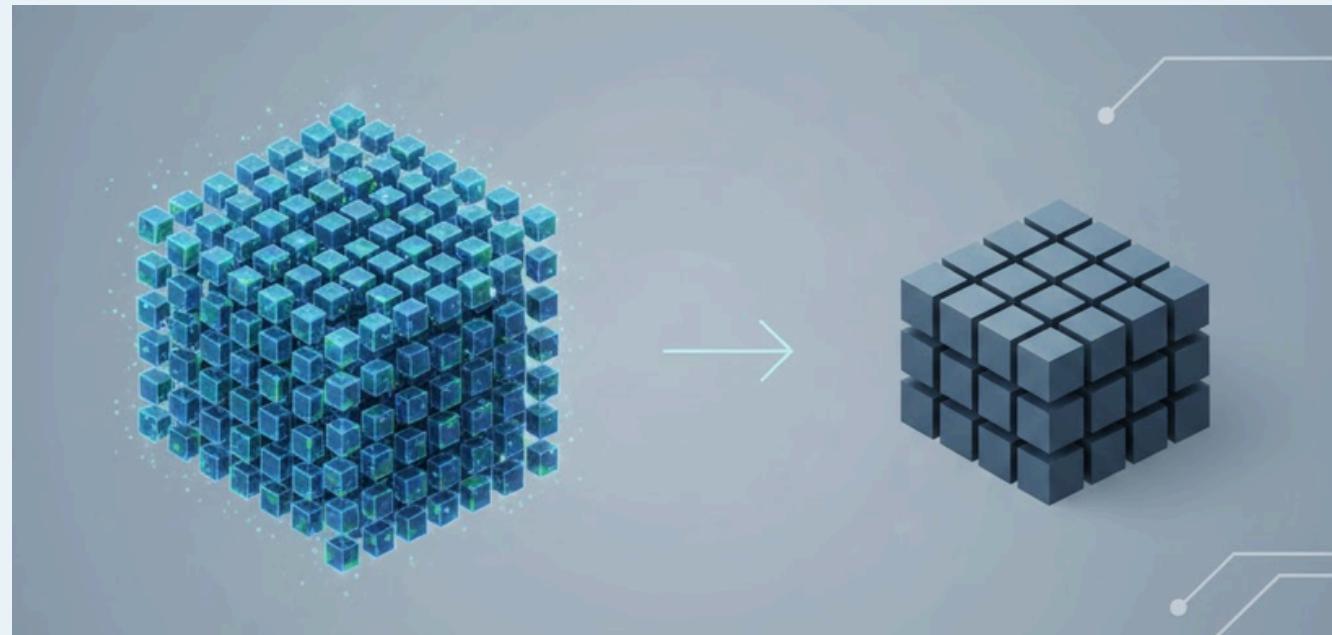
# METİNLERİN SIKIŞTIRILMASI

- Metinlerin sıkıştırılması, tekrar eden karakter ve kelimelerin daha kısa kodlarla temsil edilmesiyle yapılır ve kayıpsız sıkıştırma kullanılır.
- Amaç, metnin içeriğini bozmadan daha az yer kaplamasını sağlamaktır.



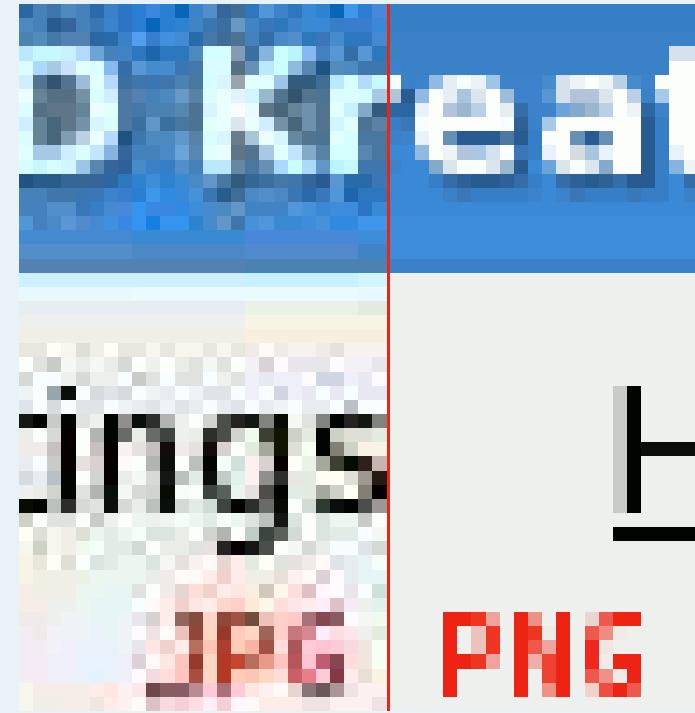
# SAYISAL VERİLERİN SIKIŞTIRILMASI

- Sayısal verilerin sıkıştırılması, tekrarlı veya öngörülebilir sayı desenlerinin daha az bit ile temsil edilmesiyle yapılır ve genellikle kayıpsız sıkıştırma tercih edilir.
- Amaç, hesaplamaları etkilemeden depolama ve iletim maliyetini düşürmektir.



# RESİMLERİN SIKIŞTIRILMASI

- Resimlerin sıkıştırılmasında, piksellerin renk değerleri doğrudan saklanmak yerine benzer renklerin gruplanması, tekrar eden desenlerin tek kez kodlanması ve insan gözünün ayırt edemediği küçük farkların atılması yoluyla daha az bit kullanılarak temsil edilir.



# **SIKIŞTIRMA SONRASI, HATA TESPİT ETME VE DÜZELTME**

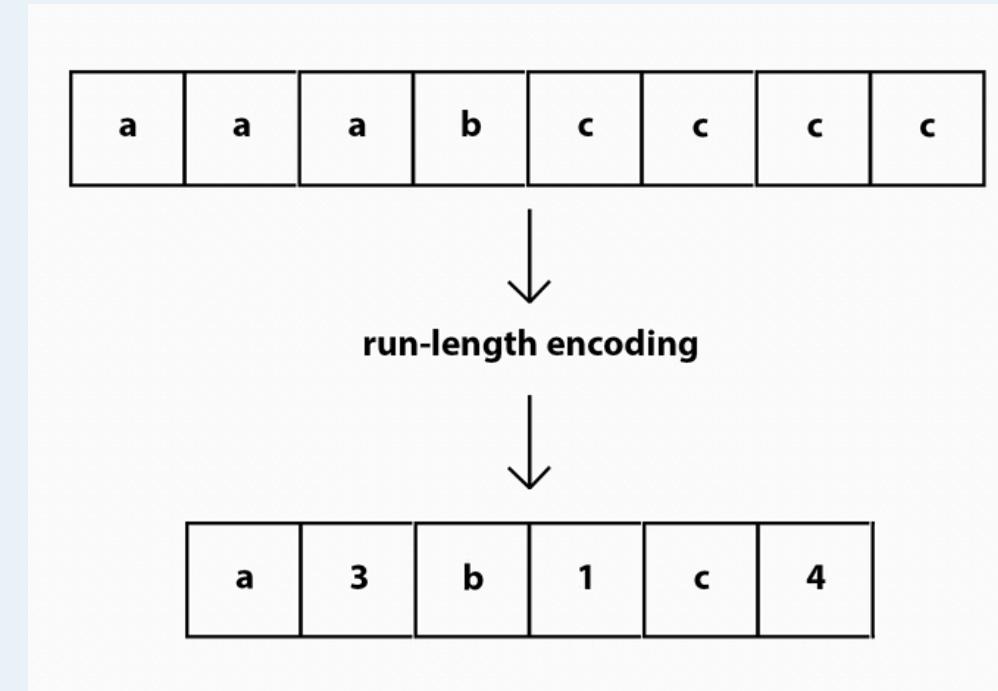
- Sıkıştırma sonrası hata tespit etme ve düzeltme, verinin depolama ya da iletim sırasında bozulup bozulmadığını kontrol etmek ve oluşan hataları tespit etmek veya düzeltmek için kullanılan yöntemleri kapsar.
- Sıkıştırılmış veriler daha az bit içerdigi için tek bir bit hatası, açma (decompression) işlemini tamamen bozabilir.
- Bu yüzden veriye hata tespit kodları (parite biti, CRC) eklenecek hatalar fark edilir;
- hata düzeltme kodları (örneğin Hamming kodu) ile belirli hatalar otomatik olarak düzeltilebilir.
- Gerçek: Sıkıştırma alan kazandırır ama dayanıklılığı azaltır; hata kontrolü yoksa veri güvenilir değildir.

# GENEL SIKIŞTIRMA TEKNİKLERİ

- Genel sıkıştırma teknikleri, verideki tekrarları, olasılık farklarını veya ardışık benzerlikleri kullanarak daha az bit ile temsil etmeyi amaçlayan yöntemlerdir.
  - RLE (Run-Length Encoding)
  - Huffman Encoding
  - Relative / Differential Encoding
  - LZW (Lempel-Ziv-Welch)

# RLE (RUN-LENGTH ENCODİNG)

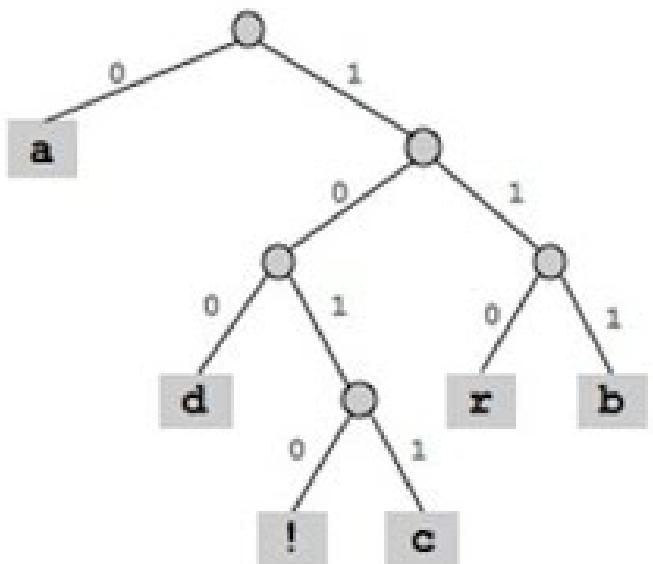
- RLE (Run-Length Encoding), ardışık olarak tekrar eden aynı veri değerlerini (değer, tekrar sayısı) şeklinde kodlayarak sıkıştırma yapan kayıpsız bir yöntemdir.
- Özellikle tek renkli alanların fazla olduğu verilerde (basit resimler, ikonlar) etkilidir; ancak veri sık tekrar içermiyorsa sıkıştırma sağlamaz.



# HUFFMAN ENCODİNG

- Huffman Encoding, veride sık geçen sembollere kısa, seyrek geçen sembollere uzun bit kodları atayarak yapılan kayıpsız bir sıkıştırma yöntemidir.
- Bu kodlar, ön ek (prefix) özelliği sayesinde açılırken karışıklık oluşturmaz.

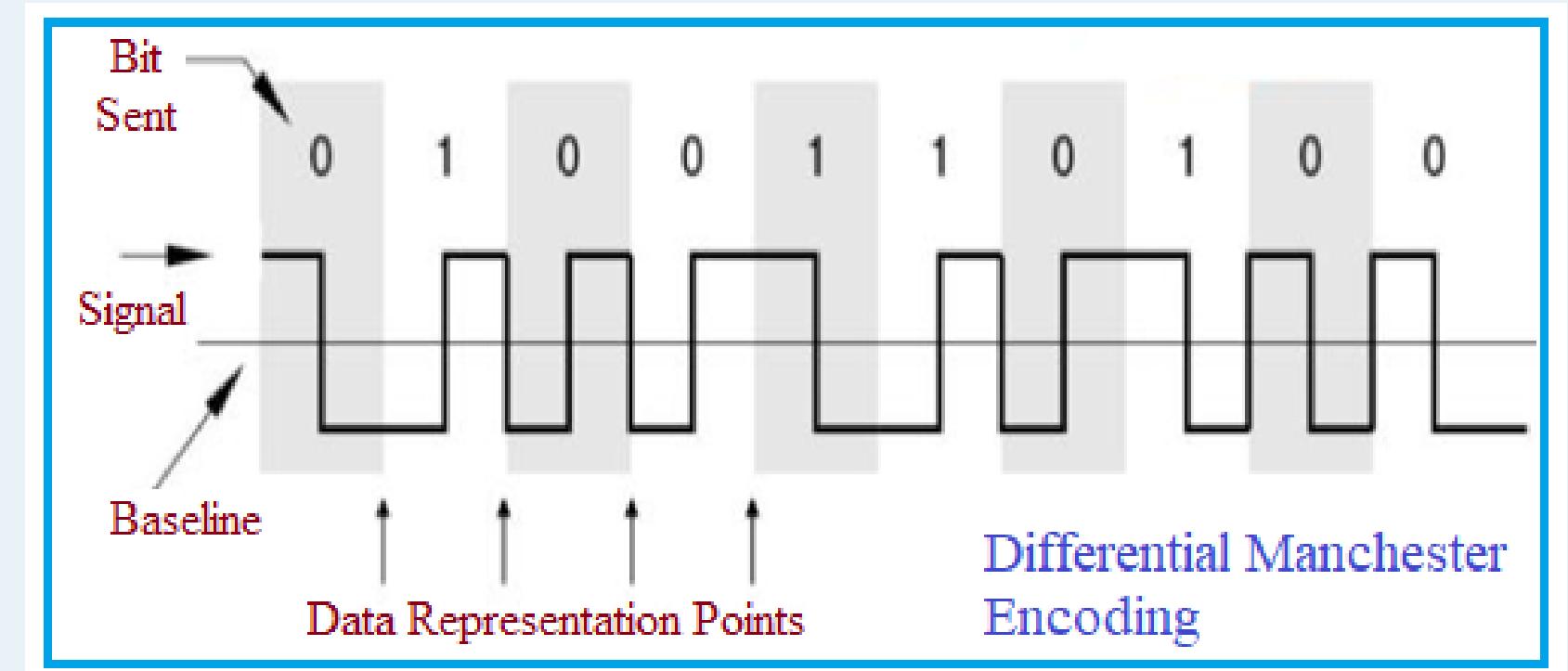
## HUFFMAN CODE DATA COMPRESSION



char	encoding
a	0
b	111
c	1011
d	100
x	110
!	1010

# RELATIVE / DIFFERENTIAL ENCODING

- Relative / Differential Encoding, ilk veri değerini doğrudan, sonraki değerleri ise bir önceki değerle arasındaki fark (delta) olarak saklayan kayıpsız bir sıkıştırma tekniğidir.
- Ardışık veriler arasındaki değişim küçükse bu farklar daha az bit ile temsil edilebilir; bu yüzden sensör verileri, ses ve zaman serileri gibi yumuşak değişen verilerde etkilidir.



# LZW (LEMPEL-ZIV-WELCH)

- LZW (Lempel-Ziv-Welch), veride tekrar eden karakter veya bit dizilerini tespit ederek bunları dinamik olarak oluşturulan bir sözlükte saklayan kayıpsız bir sıkıştırma algoritmasıdır.
- Veri işlendikçe sözlük genişler ve daha önce görülen diziler tek bir kodla temsil edilir; bu sayede dosya boyutu küçülür.

Örnek:

Girdi verisi: ABABABAB

LZW çalışırken:

İlk başta sözlükte tek karakterler vardır: A, B

Okudukça tekrar eden dizileri sözlüğe ekler:

AB, sonra ABA, sonra ABAB ...

Sonuç: ABABABAB yerine daha az sayıda kod yazılır ve veri sıkıştırılmış olur.

# SON

***“Güç, daha fazla veri tutmakta değil; daha azıyla doğrulu koruyabilmektedir.”***

# KAYNAKÇA

<https://en.wikipedia.org/wiki/ASCII>

[https://en.wikipedia.org/wiki/Character\\_encoding?utm\\_source=chatgpt.com](https://en.wikipedia.org/wiki/Character_encoding?utm_source=chatgpt.com)

<https://en.wikipedia.org/wiki/Unicode>

<https://en.wikipedia.org/wiki/UTF-8>

[https://en.wikipedia.org/wiki/Data\\_compression](https://en.wikipedia.org/wiki/Data_compression)

<https://chatgpt.com>

<https://codeguppy.com/blog/what-is-data-compression/index.html>

[https://en.wikipedia.org/wiki/Run-length\\_encoding](https://en.wikipedia.org/wiki/Run-length_encoding)

<https://www.geeksforgeeks.org/computer-networks/lzw-lempel-ziv-welch-compression-technique/>

<https://medium.com/@kutluisa.10/s%C4%B1k%C4%B1%C5%9Ft%C4%B1rma-algoritmalar%C4%B1-compression-algorithms-89c95613d9d8>

# SON

***“Güç, daha fazla veri tutmakta değil; daha azıyla doğrulu koruyabilmektedir.”***