Representing Text

- Representation of text predates the use of computers for text
- Text representation was needed for communication equipment
- One particular commonly used communication equipment was teleprinter (also called as teletypewriter, teletype or tty)

Representing Text

- Teleprinter was an electromechanical typewriter that can be used to send and receive typed messages over various types of communications channels
- Teleprinters were adapted to provide a user interface to early mainframe computers and minicomputers
- They were used for sending typed data to the computer and printing the response from the computer

Representing Text

- Computers can only understand numbers

Teleprinters and ASCII

- Thus, we need a number representation of characters: "a" or "@"
- A number representation for the English alphabet was developed in 1967 by the American Standards Association (ASA) in the US, which remains in use today
- This representation is called ASCII which stands for American Standard Code for Information Interchange

ASCII

- ASCII was originally based on the English alphabet
- ASCII encodes 128 characters into 7-bit binary integers
- 33 of these 128 are non-printing control characters

American Standard Code for Information Interchange

- "a" is encoded as $(110\ 0001)_2 = (61)_{16} = (97)_{10}$
- "@" is encoded as $(100\ 0000)_2 = (40)_{16} = (64)_{10}$

USASCII code chart

b_4	b_3	b_2	b_1	Row \ Column	0 (000)	1 (001)	2 (010)	3 (011)	4 (100)	5 (101)	6 (110)	7 (111)
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

American Standard Code for Information Interchange

- The first 32 (0 to 31) are control codes
- For example:
- Code 10 (0A) represents the "line feed" function (which causes the printer to advance its paper)
- Code 8 (08) represents "backspace"
- Code 127 (FF) represents "delete"
- Code 32 (20) represents " " (Space)

American Standard Code for Information Interchange

- The codes 33-47, 58-64, and 91-96 are printable characters
- The codes 48-57 are numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- The codes 65-90 are upper case letters: A, B, C, D, E, F, G, H, I, . . .
- The codes 97-122 are lower case letters: a, b, c, d, e, f, g, h, i, . . .

American Standard Code for Information Interchange

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	[ENG 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]

American Standard Code for Information Interchange

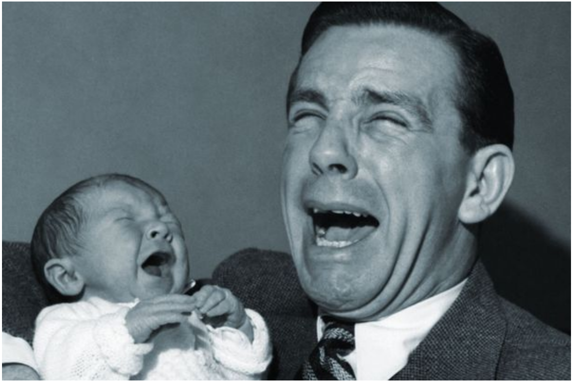
- Normally a 7-bit code is placed inside an 8-bit (1-byte) binary number, keeping the most significant (leftmost) bit 0
- Therefore, a simple text file containing the word `hello` will have 5 bytes for the 5 letters
- There may also be 1 byte space for the LF (line feed) character at the end of each line
- Demonstrate `file5.txt` and `file6.txt`

Extensions of ASCII

- Soon after ASCII was introduced, Europeans became very sad

Extensions of ASCII

- Soon after ASCII was introduced, Europeans became very sad



- They wanted codes for their alphabet characters

Extensions of ASCII

- They needed code for representation of characters found in Western European languages, such as

ç ü é â ä å ...

Extensions of ASCII

- Since we have 1 byte (8 bits), we have room for 256 characters
- This gives us 8-bit ASCII, codes from 00 (hex) to FF (hex)
- This created Extended ASCII
- Code 128 Ç
- Code 129 ü
- Code 130 é
- Code 135 ç
- Code 148 ö
- Code 153 Ö
- Code 154 Ö
- Extended ASCII does not have codes for ğ Ğ ı İ ş Ş

Extensions of ASCII

- Unfortunately, everyone had their idea of what is needed, and many standards bodies developed different variants of ASCII
- Different operating systems, game consoles, and computers have designed their own ASCII extensions
- However all of these variants include 7-bit ASCII (sometimes called US-ASCII) as their subset
- The International Organization for Standardization (ISO) standardized 8-bit codes and named is ISO 8859 in 1998
- These form **different** sets of 8-bit codes

ISO 8859

- The standard ISO 8859 covers an almost complete list of Western European languages
- It supports an extended list of languages
  - Latin-1 (Western European languages)
  - Latin-2 (Non-Cyrillic Central and Eastern European languages)
  - Latin-3 (Southern European languages and Esperanto)
  - Latin-5 (Turkish)
  - Latin-6 (Northern European and Baltic languages)
  - 8859-5 (Cyrillic)
  - 8859-6 (Arabic)
  - 8859-7 (Greek)
  - 8859-8 (Hebrew)
- **Not all software can parse ISO-8859 character sets**

Further Extension of ASCII

- Moreover, there are many more languages and scripts in the world
- Consider: Arabic, Armenian, Hebrew, Chinese, Japanese, Korean
- We need a much more comprehensive encoding system

Unicode

- Unicode is a computing industry standard for the consistent encoding, representation, and handling of text expressed in most of the world's writing systems
- Unicode Goal: **One encoding for all scripts of the world!**
- Unicode contains a repertoire of more than 110,000 characters covering 123 scripts and multiple symbol sets
- Unicode covers almost all scripts in current use today
- Unicode defines 1,114,112 code points, in the range 0 to 10FFFF

Unicode

- Unicode can be implemented by different character encodings
- The most commonly used encoding is UTF-8
- UTF stands for "Unicode Transformation Format"
- UTF-8 uses one byte for any ASCII character, all of which have the same code values in both UTF-8 and ASCII encoding, and up to four bytes for other characters

- Of more than a million code points, about 100,000 are assigned
- Most assignments are in the first 65,536 code points

Unicode Properties

- Every letter in every alphabet is assigned a number by the Unicode consortium which is written like this: U+0639
- This number is called a **code point**
- The U+ means "Unicode" and the numbers are hexadecimal
- U+0639 is the Arabic letter Ain
- The English letter A would be U+0041.
- There is no real limit on the number of letters that Unicode can define and they go beyond 65,536

Unicode Examples

- English text looks exactly the same in UTF-8 as it does in ASCII
- Specifically, `Hello` will be stored as 48 65 6C 6C 6F, which is the same as it is stored in ASCII
- The string `Hello` in Unicode corresponds to five code points: U+0048 U+0065 U+006C U+006C U+006F
- The encoding scheme determines how these bytes are to be stored
- As we said, the most commonly used encoding is UTF-8
- On the other hand , Arabic, Armenian or other letters will be represented according to their (mostly 2-byte) Unicode definitions, found in `http://www.unicode.org/charts`

Unicode 7-bit ASCII in Python

- In Python, all strings are stored in UTF-8 Unicode
- 7-bit ASCII (US-ASCII) Example:

```
>>> s = "Hello"
>>> for c in s:
        print(hex(ord(c)))


0x48
0x65
0x6c
0x6c
0x6f
>>>
```

Unicode 8-bit ASCII in Python

```
>>> s = "weiß Köln"
>>> for c in s:
        print(hex(ord(c)))
```

```
0x77
0x65
0x69
0xdf
0x20
0x4b
0xf6
0x6c
0x6e
```

Unicode Arabic

```
>>> chr(0x06A0)
'ڠ'
>>> chr(0x06A1)
'ڡ'
>>> chr(0x06A2)
'ڢ'
>>> for i in range(0,17):
        print(chr(0x6A0+i), end="")
```

ڠڡڢڣڤڥڦڧڨکڪګڬڭڮگڰ

Unicode Armenian

```
>>> chr(0x0531)
'Ա'
>>> chr(0x0532)
'Բ'
>>> chr(0x0533)
'Գ'
>>> for i in range(1,17):
        print(chr(0x530+i), end="")

ԱԲԳԴԵԶԷԸԹԺԻԼԽԾԿՀ
```

Unicode CJK - Chinese Japanese Korean

```
>>> chr(0x4e78)
'乸'
>>> chr(0x4e79)
'乹'
>>> chr(0x4e7a)
'乺'
>>> for i in range(20):
        print(chr(0x4e78+i), end="")
```

乸乹乺乻乼乽乾乿亀亁亂亃亄亅了亇予争亊事

Unicode CJK - Chinese Japanese Korean

```
>>> for i in range(12,32):
        for j in range(32):
                print(chr(0x4fa0+i*32+j), end="")
        print()
```

儠儡儢儣儤儥儦儧儨儩優儫儬儭儮儯儰儱儲儳儴儵儶儷儸儹儺儻儼儽儾儿
兀允兂元兄充兆兇先光兊克兌免兎兏児兑兒兓兔兕兖兗兘兙党兛兜兝兞兟
兠兡兢兣兤入兦內全兩兪八公六兮兯兰共兲关兴兵其具典兹兺养兼兽兾兿
冀冁冂冃冄内円冇冈冉冊冋册再冎冏冐冑冒冓冔冕冖冗冘写冚军农冝冞冟
冠冡冢冣冤冥冦冧冨冩冪冫冬冭冮冯冰冱冲决冴况冶冷冸冹冺冻冼冽冾冿
净凁凂凃凄凅准凇凈凉凊凋凌凍凎减凐凑凒凓凔凕凖凗凘凙凚凛凜凝凞凟
几凡凢凣凤凥処凧凨凩凪凫凬凭凮凯凰凱凲凳凴凵凶凷凸凹出击凼函凾凿
刀刁刂刃刄刅分切刈刉刊刋刌刍刎刏刐刑划刓刔刕刖列刘则刚创刜初刞刟
删刡刢刣判別刦刧刨利刪别刬刭刮刯到刱刲刳刴刵制刷券刹刺刻刼刽刾刿
剀剁剂剃剄剅剆則剈剉削剋剌前剎剏剐剑剒剓剔剕剖剗剘剙剚剛剜剝剞剟
剠剡剢剣剤剥剦剧剨剩剪剫剬剭剮副剰剱割剳剴創剶剷剸剹剺剻剼剽剾剿
劀劁劂劃劄劅劆劇劈劉劊劋劌劍劎劏劐劑劒劓劔劕劖劗劘劙劚力劜劝办功
加务劢劣劤劥劦劧动助努劫劬劭劮劯劰励劲劳労劵劶劷劸効劺劻劼劽劾势
勀勁勂勃勄勅勆勇勈勉勊勋勌勍勎勏勐勑勒勓勔動勖勗勘務勚勛勜勝勞募
勠勡勢勣勤勥勦勧勨勩勪勫勬勭勮勯勰勱勲勳勴勵勶勷勸勹勺勻勼勽勾勿
匀匁匂匃匄包匆匇匈匉匊匋匌匍匎匏匐匑匒匓匔匕化北匘匙匚匛匜匝匞匟
匠匡匢匣匤匥匦匧匨匩匪匫匬匭匮匯匰匱匲匳匴匵匶匷匸匹区医匼匽匾匿
區十卂千卄卅卆升午卉半卋卌卍华协卐卑卒卓協单卖南単卙博卛卜卝卞卟
占卡卢卣卤卥卦卧卨卩卪卫卬卭卮卯印危卲即却卵卶卷卸卹卺卻卼卽卾卿
厀厁厂厃厄厅历厇厈厉厊压厌厍厎厏厐厑厒厓厔厕厖厗厘厙厚厛厜厝厞原

Unicode CJK - Chinese Japanese Korean

儠儡儢儣儤儥儦儧儨儩優儫儬儭儮儯儰儱儲儳儴儵儶儷儸儹儺儻儼儽儾儿
兀允兂元兄充兆兇先光兊克兌免兎兏児兑兒兓兔兕兖兗兘兙党兛兜兝兞兟
兠兡兢兣兤入兦內全兩兪八公六兮兯兰共兲关兴兵其具典兹兺养兼兽兾兿
冀冁冂冃冄内円冇冈冉冊冋册再冎冏冐冑冒冓冔冕冖冗冘写冚军农冝冞冟
冠冡冢冣冤冥冦冧冨冩冪冫冬冭冮冯冰冱冲决冴况冶冷冸冹冺冻冼冽冾冿
净凁凂凃凄凅准凇凈凉凊凋凌凍凎减凐凑凒凓凔凕凖凗凘凙凚凛凜凝凞凟
几凡凢凣凤凥処凧凨凩凪凫凬凭凮凯凰凱凲凳凴凵凶凷凸凹出击凼函凾凿
刀刁刂刃刄刅分切刈刉刊刋刌刍刎刏刐刑划刓刔刕刖列刘则刚创刜初刞刟
删刡刢刣判別刦刧刨利刪别刬刭刮刯到刱刲刳刴刵制刷券刹刺刻刼刽刾刿
剀剁剂剃剄剅剆則剈剉削剋剌前剎剏剐剑剒剓剔剕剖剗剘剙剚剛剜剝剞剟
剠剡剢剣剤剥剦剧剨剩剪剫剬剭剮副剰剱割剳剴創剶剷剸剹剺剻剼剽剾剿
劀劁劂劃劄劅劆劇劈劉劊劋劌劍劎劏劐劑劒劓劔劕劖劗劘劙劚力劜劝办功
加务劢劣劤劥劦劧动助努劫劬劭劮劯劰励劲劳労劵劶劷劸効劺劻劼劽劾势
勀勁勂勃勄勅勆勇勈勉勊勋勌勍勎勏勐勑勒勓勔動勖勗勘務勚勛勜勝勞募
勠勡勢勣勤勥勦勧勨勩勪勫勬勭勮勯勰勱勲勳勴勵勶勷勸勹勺勻勼勽勾勿
匀匁匂匃匄包匆匇匈匉匊匋匌匍匎匏匐匑匒匓匔匕化北匘匙匚匛匜匝匞匟
匠匡匢匣匤匥匦匧匨匩匪匫匬匭匮匯匰匱匲匳匴匵匶匷匸匹区医匼匽匾匿
區十卂千卄卅卆升午卉半卋卌卍华协卐卑卒卓協单卖南単卙博卛卜卝卞卟
占卡卢卣卤卥卦卧卨卩卪卫卬卭卮卯印危卲即却卵卶卷卸卹卺卻卼卽卾卿
厀厁厂厃厄厅历厇厈厉厊压厌厍厎厏厐厑厒厓厔厕厖厗厘厙厚厛厜厝厞原