ASCII Character Set
A Way of Representing Textual Data

(American Character Set for Information InterChange)

Data Representation

There are different forms of data - these include text, image , sound, video.

The Computer uses different ways of representing these forms of data.

We will look at how the computer represents textual data.

Text

(1) A text document can be decomposed into:

Paragraphs

Sentences

Words

And finally individual characters

(2) To represent a text document in digital form (i.e in a form intelligible to the computer, you need to represent each character that you can come across.

(3) All of these characters need to be represented and stored in computer memory.

(4) There are a finite number of characters to represent.

Standard ASCII Character Set

Characters can be represented by the Standard ASCII Character Set.

ASCII is an acronym for American Standard Code for Information Interchange.

In ASCII each character has a decimal number associated with it.

Then this decimal number is converted to binary.

This binary number is stored in the computer.

Let us see examples of this.

Standard ASCII Character Set

A = 65 = 1000 001

Where
A is the Character
65 is the decimal number.
1000 001 is the binary number

Note there are 7 bits in the binary number.

In the Standard ASCII system, 7 bits are allocated to represent a character.

64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1
1	1	1	1	1	1	1
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1

B = 66 = 1000 010

Where
B is the Character
66 is the decimal number.
1000 010 is the binary number

C = 67 = 1000 011

Where
C is the Character
67 is the decimal number.
1000 011 is the binary number

Question:
How do you think the character ‘Z’ will be represented?

Character Set

B = 66 = 1000 010

Where
B is the Character
66 is the decimal number.
1000 010 is the binary number

C = 67 = 1000 011

Where
C is the Character
67 is the decimal number.
1000 011 is the binary number

Question:
How do you think the character ‘Z’ will be represented?

Z = 90 = 1011 010

Where
Z is the Character
90 is the decimal number.
1000 010 is the binary number

You must learn this definition

A Character Set is a collection of characters a computer recognises them from their binary representation.

More about the Standard ASCII Character Set

The Standard ASCII Character Set has 7 bits representing each character

This means in total 128 characters can be represented (from 2 to the power 7 or 2^7)

Below is a table of the Standard ASCII Character Set

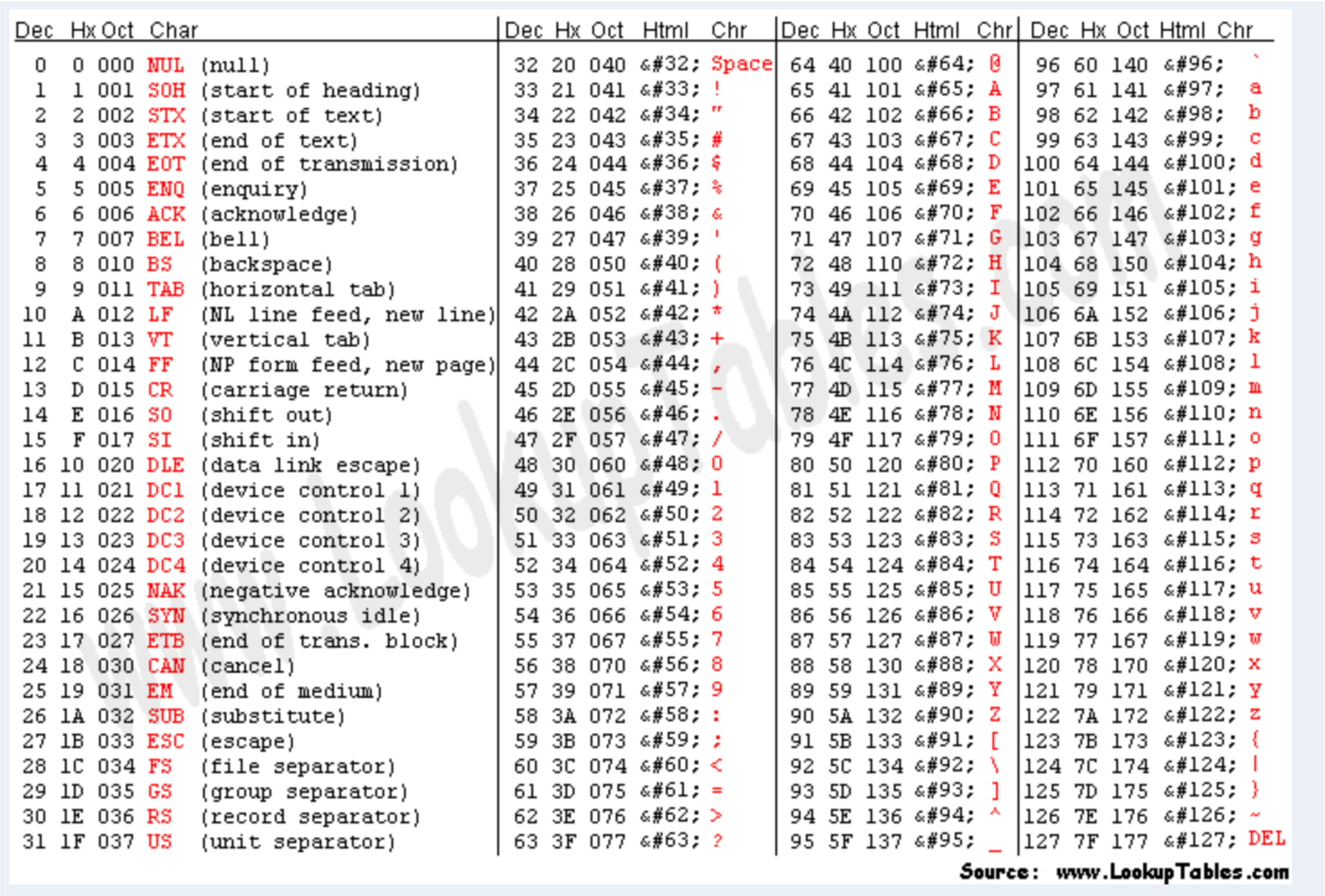
The ASCII Character Set can be divided into:-

(1) Printable Characters - these are characters which can be printed such as:-(
(2) a) Uppercase letters (A-Z)
(3) (b) Lowercase letters (a-z)
(4) (c) Numbers (0-9)

Letters and Numbers are known as alphanumeric characters.

(1) Control Characters - those that control the keyboard
(2) Eg. newline, tab, etc. (1 to 31)

Standard ASCII Character Set



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Extended ASCII Character Set

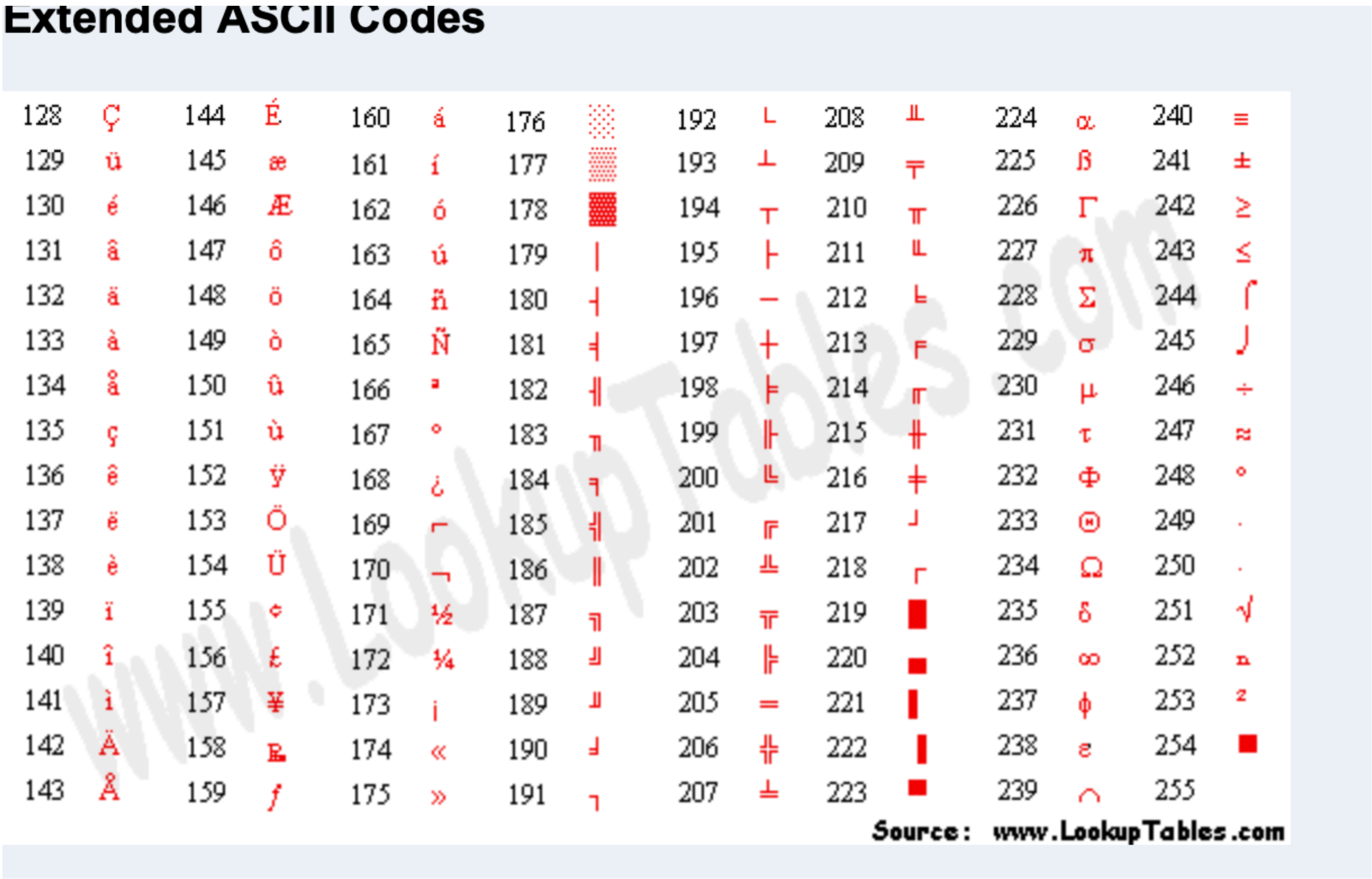
The Extended Character Set has 8 bits.

This means the Extended Character Set can represent 256 Characters (this is from 2 to the power 8 or 2^8).

128	64	32	16	8	4	2	1
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1
1	1	1	1	1	1	1	1
Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1

An extra feature that the Extended ASCII Character Set allows for is special symbols in languages.

Extended ASCII Codes



Extended ASCII Codes

128	Ç	144	É	160	á	176	░	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	ⁿ
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

Source: www.LookupTables.com

Why do we need the ASCII system?

The first computers stored characters in their own way.

This worked well until people decided that data and information should be moved and shared between computers so an universal system to represent data was required.

This led to the development of the ASCII system in 1963.

ASCII is now an universal system used the world over by all computers.

ASCII is a system in which symbols/data/code is represented.

Before ASCII different computers had no way of communicating with each other.

The ASCII system is an universal standard so computers can communicate with each other.

Programming Python Code and the ASCII System

Python Code to find the ASCII values of Characters

```
>>> ord (‘A’)

65

>>> ord (‘B’)

66

>>> ord (‘a’)
97

>>> ord (‘b’)

98
```

Python Code to find the Characters values of for given ASCII numbers

```
>>> chr (65)
‘A’

>>> chr (’66’)

‘B’
```

Disadvantages of the ASCII System

The ASCII Character System can only represent characters from the English language and it cannot represent characters from foreign languages. For that purpose we need the Unicode character set

So, that is overcome using the UNICODE Character Set.

UNICODE's origins date back to 1987 from Apple.

Some UNICODE systems have 16 bits and can represent over 65,000 characters (65, 536).
This is from 2 to the power 16 (or 2^{16})

Other UNICODE systems have 32 bits and can represent 2 to the power 32 (or 2^{32}) characters.

Unicode can handle characters from all languages in the world
And all mathematical symbols.

The Unicode System

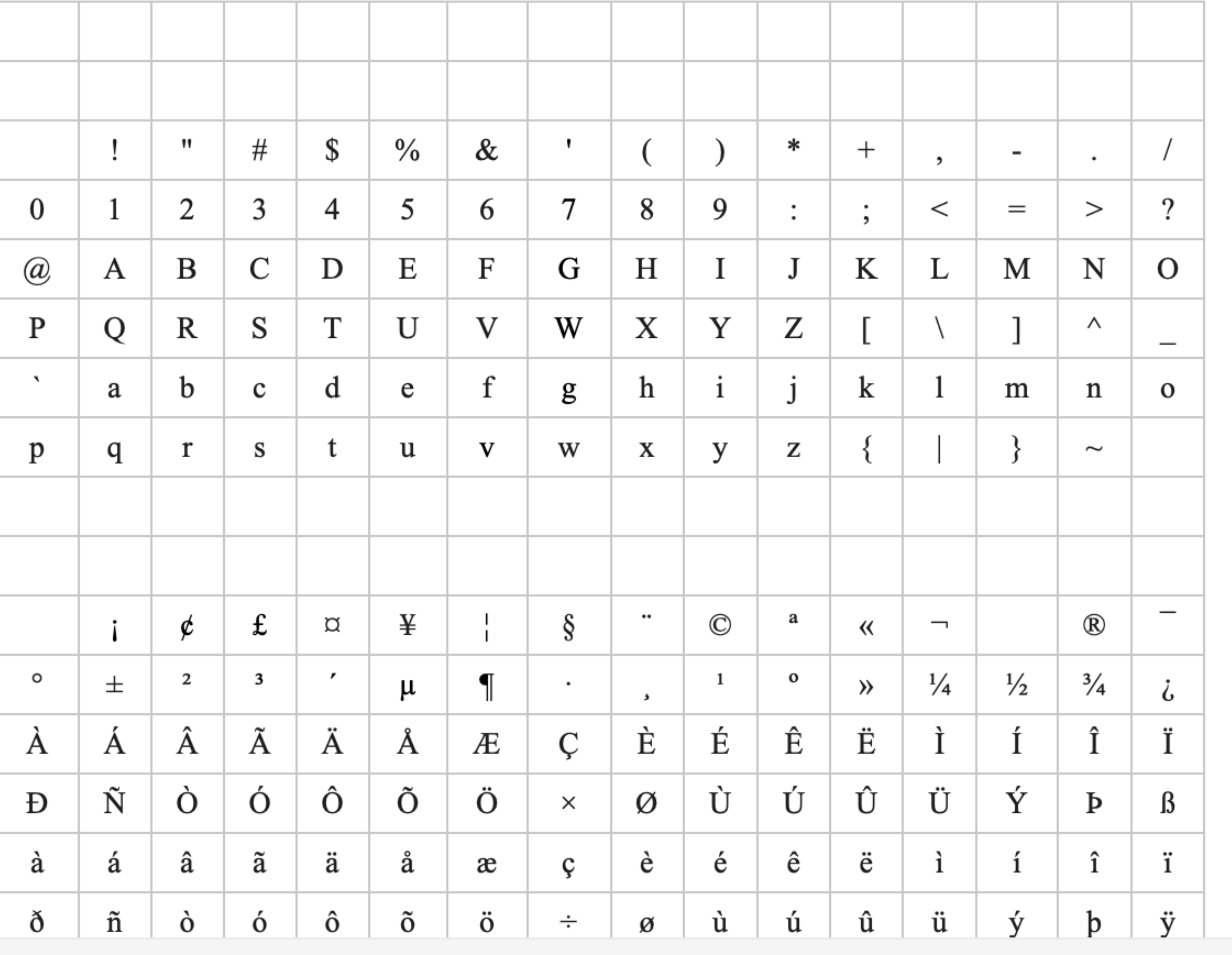
Imagine you own a company that sells computers all over the world.

Naturally, every customer will want to type in their own language. What shall we do?

So, if you want a system which can handle every possible written language - when the computer gets turned on for the first time, the customer simply chooses the language of their choice.

UNICODE can handle any language.

Some Characters that can be represented by the Unicode System



	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Questions & Answers

(1) Describe what is meant by the term 'Character Set'? (They are asking you to define the term 'Character Set')

(2) Compare the use of the Standard ASCII Character Set and the Extended ASCII Character Set to represent characters.

(3) Explain why mobile phones that can send emoji would use UNICODE instead of ASCII as their character set?

Questions

(1) Describe what is meant by the term 'Character Set'? (They are asking you to define the term 'Character Set')

A Character Set is a collection of characters a computer recognises them from their binary representation.

(2) Compare the use of the Standard ASCII Character Set and the Extended ASCII Character Set to represent characters.

As the Standard ASCII uses 7 bits to represent characters and the Extended ASCII representation uses 8 bits
To represent characters, the Extended ASCII system can represent more characters
Than the Standard ASCII system.

(3) Explain why mobile phones that can send emoji would use UNICODE instead of ASCII as their character set?

The Standard ASCII character set has 7 bits to represent a character
And the Extended Character Set has 8 bits to represent a character .
Unicode uses 16 bits to represent characters.
Therefore the Unicode system can represent more characters
And this includes the emoji symbols whereas
The ASCII system does not have enough space to represent the Emoji symbols.

Programming Questions which involve knowledge of the use of the ASCII Character Set

(1) The Caeser Cipher is an encryption method. This is how it works:-

Here, the Caesar Cipher moves each letter of the alphabet one place to the right.

The following table shows the original letters in the first row, and the new letters in the second row.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A

For example, if the message read: HELLO

This would be read as: IFMMP

Programming Questions which involve knowledge of the use of the ASCII Character Set

The following pseudocode algorithm takes a string of uppercase letters as input and uses the Caesar Cipher to encrypt them.

The functions used in the algorithm are described in the table.

Function	Description
ASC(character)	Returns the ASCII value for character e.g. ASC(“A”) returns 65
CHR(ASCIIValue)	Returns the single character for ASCIIValue e.g. CHR(65) returns “A”
subString(Value, Number)	Returns the Number of characters

Complete the pseudocode algorithm to perform a Caesar Cipher

```
message = input(“Please enter your string”)
newMessage = “”
messageLength  = message.length
for count = 0 to …………………………
    ASCIIValue  = ASC(message.subString(………………….,1))
    ASCIIValue  = ASCIIValue  + ……………………………..
    if ASCIIValue  >90  then
        ASCIIValue  = ……………….. - 26
    endif
    newMessage = ………………… + CHR(ASCIIValue)
next count
print (“……………………”)
```

Programming Questions which involve knowledge of the use of the ASCII Character Set

The following pseudocode algorithm takes a string of uppercase letters as input and uses the Caesar Cipher to encrypt them.

The functions used in the algorithm are described in the table.

Function	Description
ASC(character)	Returns the ASCII value for character e.g. ASC("A") returns 65
CHR(ASCIIValue)	Returns the single character for ASCIIValue e.g. CHR(65) returns "A"
subString(Value, Number)	Returns the Number of characters

Complete the pseudocode algorithm to perform a Caesar Cipher

```
message = input("Please enter your string")
newMessage = ""
messageLength  = message.length
for count = 0 to ….messageLength
    ASCIIValue  = ASC(message.subString(….count……..,1))
    ASCIIValue  = ASCIIValue  + ……1……………………….
    if ASCIIValue  >90  then
        ASCIIValue  = ….ASCIIValue - 26
    endif
    newMessage = ….newMessage + CHR(ASCIIValue)
next count
print ("….newMessage…….")
```

Programming Questions which involve knowledge of the use of the ASCII Character Set

The following pseudocode algorithm takes a string of uppercase letters as input and uses the Caesar Cipher to encrypt them.

The functions used in the algorithm are described in the table.

Pseudocode Algorithm to perform a Caesar Cipher

```
message = input(“Please enter your string”)
newMessage = “”
messageLength  = message.length

for count = 0 to …..messageLength
        ASCIIValue  = ASC(message.subString(……count……………..,1))
        ASCIIValue  = ASCIIValue  + ………1………………………..
        if ASCIIValue  >90  then
                ASCIIValue  = ……ASCIIValue……….. - 26
        endif
        newMessage = ……newMessage……. + CHR(ASCIIValue)
next count
print (“…newMessage….”)
```

Now write the corresponding program

Programming Questions which involve knowledge of the use of the ASCII Character Set

The following pseudocode algorithm takes a string of uppercase letters as input and uses the Caesar Cipher to encrypt them.

The functions used in the algorithm are described in the table.

Pseudocode Algorithm to perform a Caesar Cipher

```
message = input(“Please enter your string”)
newMessage = “”
messageLength  = message.length

for count = 0 to …..messageLength
        ASCIIValue  = ASC(message.subString(……count……,1))
        ASCIIValue  = ASCIIValue  + ……1……
        if ASCIIValue  >90  then
                ASCIIValue  = ……ASCIIValue…… - 26
        endif
        newMessage = ……newMessage…… + CHR(ASCIIValue)
next count
print (“……newMessage……”)
```

Now write the corresponding program. Type it in and run it.

```
message = input(“Please enter your string”)
newMessage = “”
messageLength  = len(message)
for count  in range (0 to ……messageLength)
        ASCIIValue  = ord(message[count:count+1])
        ASCIIValue  = ASCIIValue  + ……1……
        if ASCIIValue  >90 :
                ASCIIValue  = ……ASCIIValue…… - 26
        # endif
        newMessage = ……newMessage…… + chr(ASCIIValue)
#next count
print (“……newMessage……”)
```

Questions About Programming

Write a Program which generates a random password.
The password should have a random length of between 7 and 10 characters.
Each character should be randomly selected from positions 33 and 126 in the ASCII table.
Your function will not take any parameters.
It will return the randomly generated password as its only result.

Hint: You will probably find the chr function helpful when completing this question.

Questions About Programming

Write a Program which generates a random password.
The password should have a random length of between 7 and 10 characters.
Each character should be randomly selected from positions 33 and 126 in the ASCII table.
Your function will not take any parameters.
It will return the randomly generated password as its only result.

Hint: You will probably find the chr function helpful when completing this question.

```
import random

 SHORTEST = 7
 LONGEST = 10

 MIN_ASCII = 33
MAX_ASCII = 126


# Generate a Random Password
def randomPassword():
    # Select a Random Length for the password
    randomLength = random.randint(SHORTEST, LONGEST)
    print("randomLength ", randomLength)
    result = ""
    for j in range(randomLength):
        randomCharacter = chr(random.randint(MIN_ASCII,MAX_ASCII))
        print("randomCharacter ", randomCharacter)
        result =  result + randomCharacter
        print("result ", result)
    return(result)

Password = randomPassword()
print("Password is ", Password)
```

Some Output

```
randomLength  9
randomCharacter  /
result  /
randomCharacter  y
result  /y
randomCharacter  3
result  /y3
randomCharacter  &
result  /y3&
randomCharacter  _
result  /y3&_
randomCharacter  W
result  /y3&_W
randomCharacter  :
result  /y3&_W:
randomCharacter  8
result  /y3&_W:8
randomCharacter  2
result  /y3&_W:82
Password is  /y3&_W:82
```