Sending Text

Unit 1 Lesson 7 (U1.7)

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

In previous lessons we explored how to encode numbers in binary, and you also developed protocols for sending a list of numbers.

Today we’re going to take that method one step further and look at how we can encode text with a binary representation.

Hopefully you are beginning to realize that if we can figure out a way to represent information as a set of numbers, then we can encode it in bits and store that information in a computer or send it over the Internet.

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Think Pair Share - How would you encode text?

One of the most powerful uses of the internet is sending text to people. Since the internet can only send bits around we need a way to encode text with bits...

If it were up to you, **how would you encode text in binary**? Quickly, jot down an idea for encoding text in your INB.

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Did you and your neighbor come up with the exact same idea? What was different?

How many bits does your encoding scheme require? For example, how many bits would you need to say "hello"?

Did you account for anything besides the letters of alphabet (or whole words)?

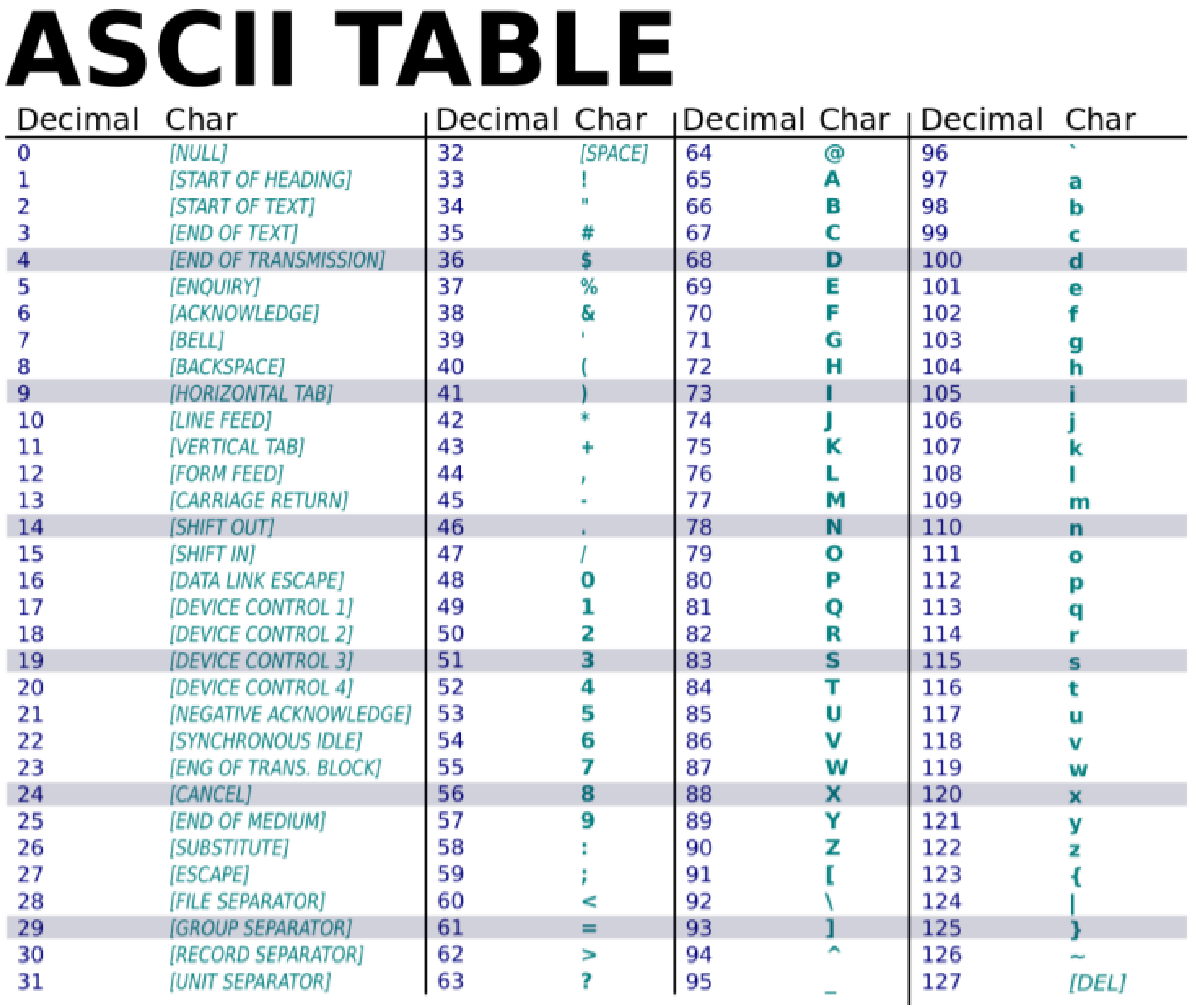
Of the encoding schemes mentioned so far, which one is "best"? why?

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

You just invented your own scheme for encoding text with numbers. It turns out that there is a standard encoding for most of the symbols you can type on an American keyboard.

That encoding is called the American Standard Code for Information Interchange or ASCII (pronounced: “Ask-ee”).

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text



ASCII TABLE

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

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

ASCII codes were originally 7 bits long and so there are 128 possible values.

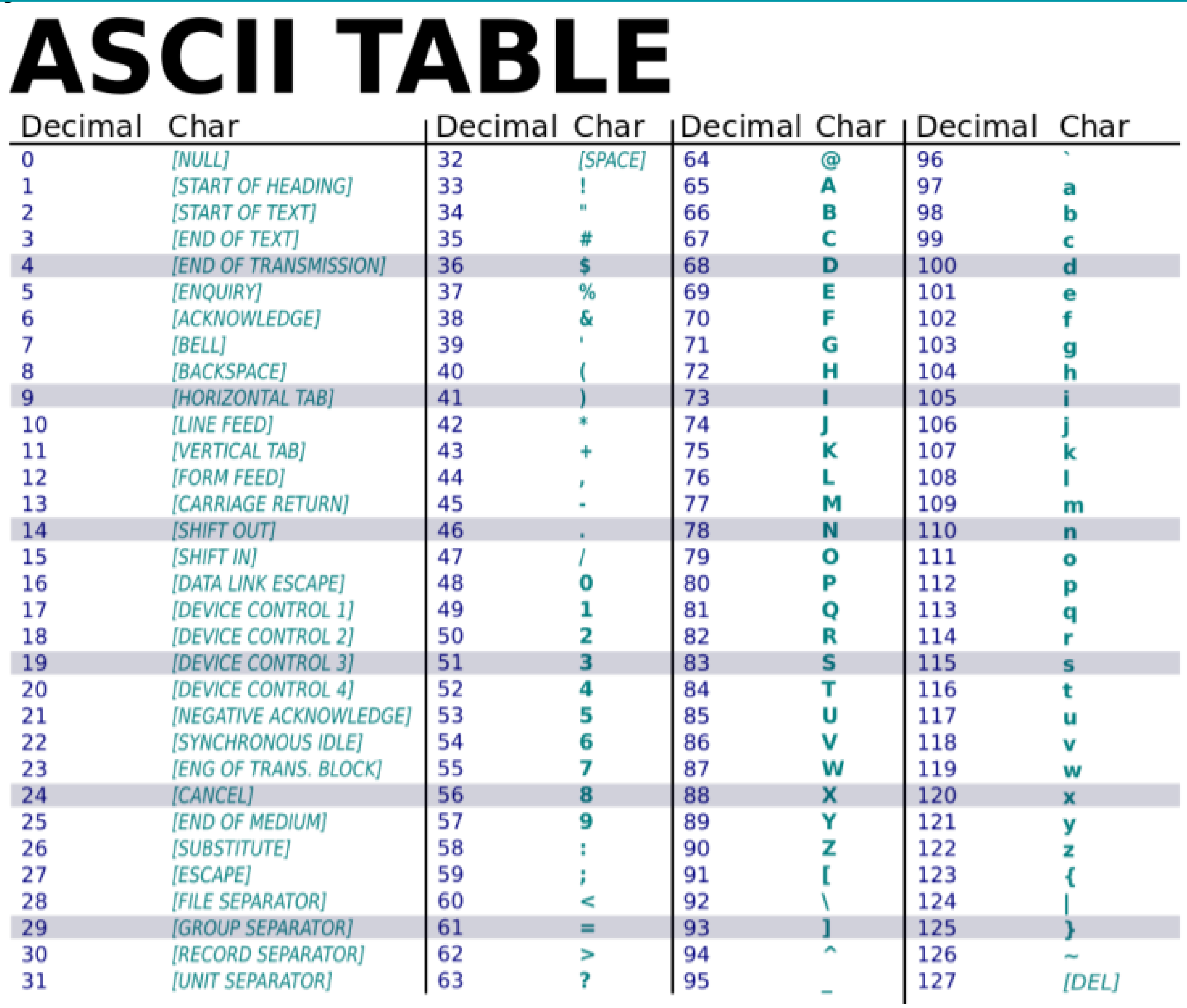
0-31 are “control characters” that are largely defunct and go unused; they were formerly used to control various aspects of machines and printers.

32-126 are printable characters and include the numbers 0-9, all 26 letters (both lowercase and uppercase), and many common punctuation symbols.

127 is the symbol for delete.

Over time, 8 bits became a standard “chunk-size” for encoding information. ASCII made the transition to this 8-bit encoding by just adding an extra 0 to the front of the old 7-bit codes.

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text



ASCII TABLE

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

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Quick Activity: write your name in ASCII codes

Using the ASCII table, translate your first name from letters to numbers using the ASCII table.

Write your name as:

Name!

(capital first letter, exclamation point at the end)

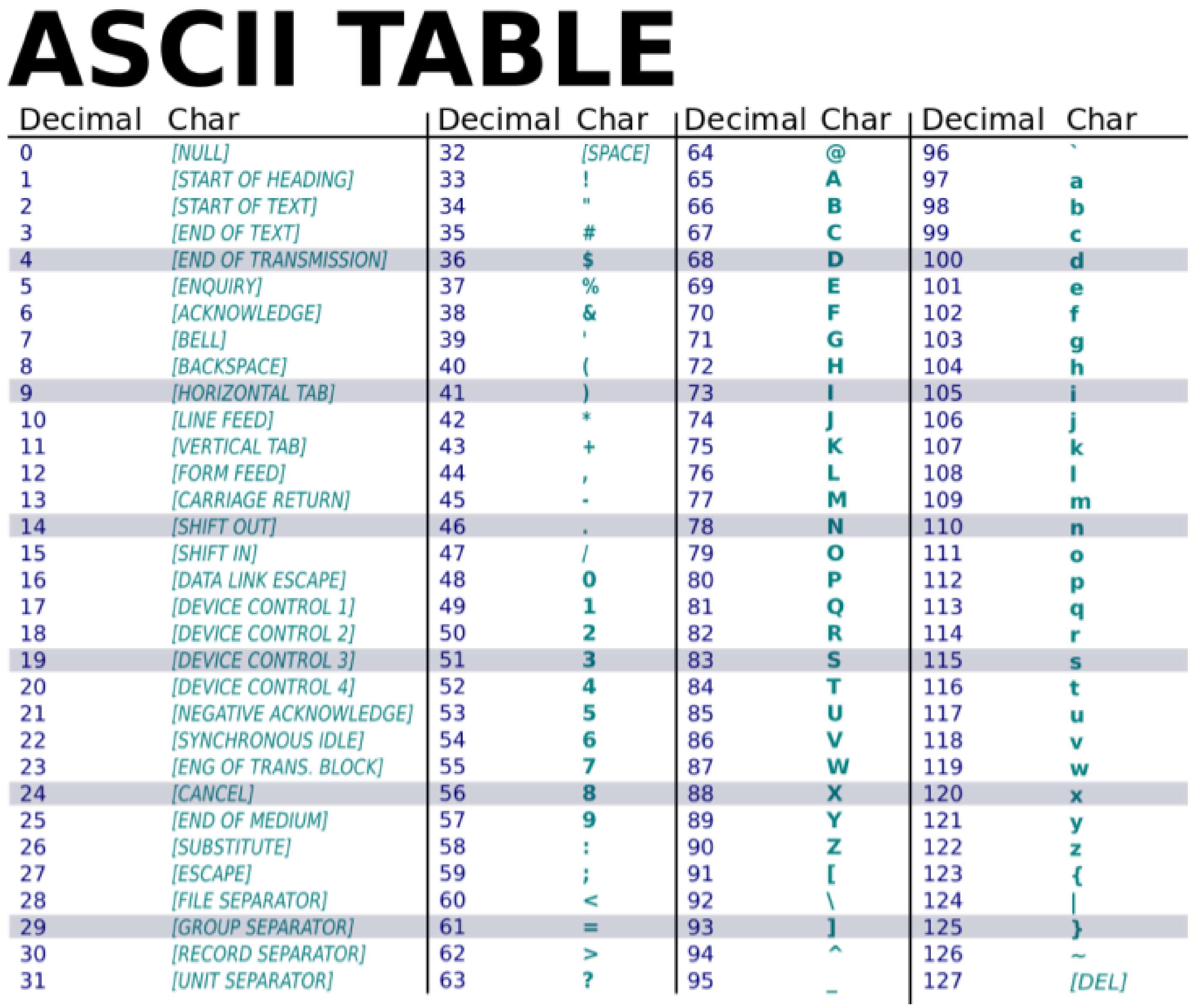
Check your name here.

(post in Classroom)

77 114 46 32 75 101 114 115 104 97 119

Mr. Kershaw

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text



ASCII TABLE

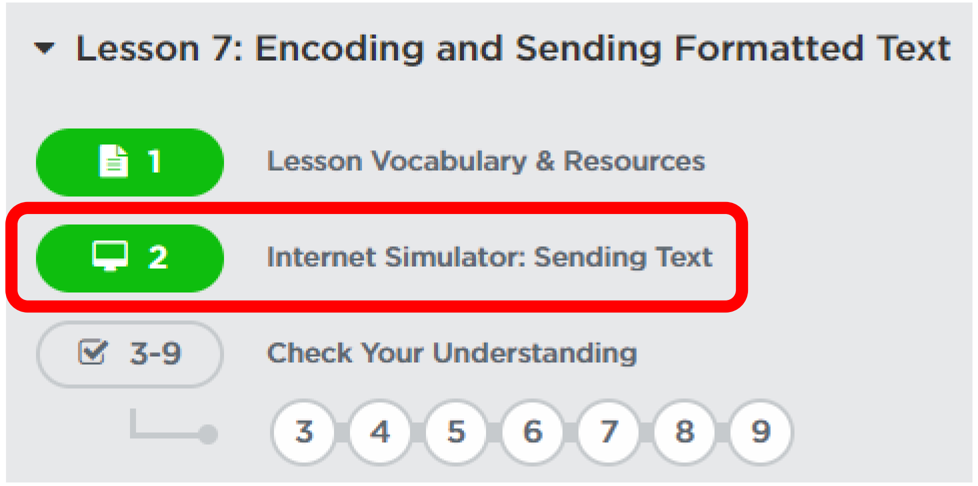
Decimal	Char	Decimal	Char	Decimal	Char	Decimal	Char
0	[NULL]	32	[SPACE]	64	@	96	`
1	[START OF HEADING]	33	!	65	A	97	a
2	[START OF TEXT]	34	"	66	B	98	b
3	[END OF TEXT]	35	#	67	C	99	c
4	[END OF TRANSMISSION]	36	$	68	D	100	d
5	[ENQUIRY]	37	%	69	E	101	e
6	[ACKNOWLEDGE]	38	&	70	F	102	f
7	[BELL]	39	'	71	G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

CODE

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Activity: Formatting Text Challenge

Go to Lesson 7, navigate to the Internet Simulator.


Lesson 7: Encoding and Sending Formatted Text

1 Lesson Vocabulary & Resources

2 Internet Simulator: Sending Text

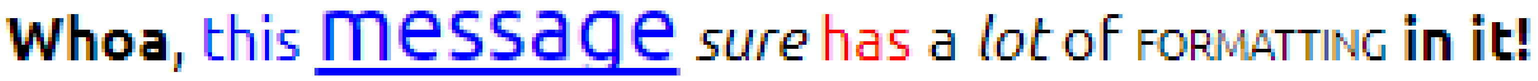
3-9 Check Your Understanding

3 4 5 6 7 8 9

Connect with your partner.
Explore the NEW Internet Simulator and see what's new and improved.
(4 mins)

What's new?

Can you send this message with the Internet Simulator?


Whoa, this message sure has a *lot* of FORMATTING **in it!**

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Activity: Formatting Text Challenge

Both the text and the formatting instructions must be derived from the printable ASCII character set (i.e. codes 32-126).

Challenge #1 (7 mins)
Create a protocol, in your INB, that allows you to send and a receive a message that contains **BOLD**, *ITALICS*, and UNDERLINING

Challenge #2 (4 mins)
Expand your protocol to allow you to receive a message that contains three different font sizes.

(LARGE, MEDIUM, SMALL)

Challenge #3 (4 mins)
Now, include rules for font color.
(RED, BLACK, BLUE)

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Activity: Formatting Text Challenge

Now that you have your protocol for the 3 challenges. Move to the opposite side of the room or to a place where you can not see what is on your partner's desk.

I will give you a message to send to your partner. Using your protocol, send your message and have them decode the it. Once decoded, have them show the message to you to verify if they decoded the message correctly.

Wrap-Up

Was your group successful?

If not, what caused the most trouble?

Were some components of the challenge easier to address than others?

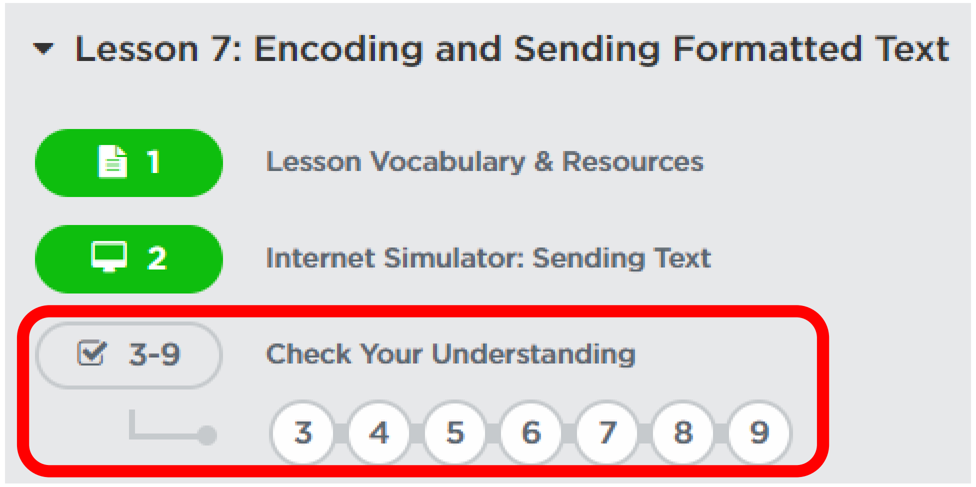
Think-Pair-Share

Take a moment to think about the layers of encodings that allowed for formatted text to be transmitted over the Internet.

Imagine someone pointed to piece of formatted text and asked: 'Can you explain to me how this is encoded in binary?' How would you explain it?

Unit 1 Lesson 7 (U1.7)
Encoding and Sending Formatted Text

Complete the Check for Understanding



Lesson 7: Encoding and Sending Formatted Text

1
Lesson Vocabulary & Resources
2
Internet Simulator: Sending Text
3-9
Check Your Understanding
3
4
5
6
7
8
9