Unit 181
Baudot encoding

During and after World War II, texting ("SMS" ["short-message service"] for European readers) was done with large devices, called "teleprinters," that did not fit easily into one's pocket. Bits were very expensive then, and most users could not afford a complete byte of eight bits. Therefore, teleprinters used five parallel transmission lines, and a means of encoding characters with five bits (voltage on and off for each line) was needed. By this time, however, punctuation and digits had been invented, and message senders needed more than the thirty-two possibilities afforded by five bits. Hence, the symbol and letter shifts were invented. A special character, called *symbol shift*, changes the character set to one of digits and punctuation. Another special character, the *letter shift*, returns the teleprinter to the character set containing letters. Upper-case letters were all that existed, so there was no such thing as that sort of shift key. The entire scheme, including the assignments of bit patterns to characters, is called *Baudot* encoding. In Table 181.1 we see the bit patterns and the characters assigned to them for both letter and symbol mode. In addition, we see the notation used by cryptanalysts at Bletchley Park during World War II as they worked to break the cipher machines that used it. These machines used innovative ways to flip and/or rearrange bits with rotors. Unlike the Enigma and such machines, in which the rotors defined permutations of the alphabet, these new machines used rotors to control the flipping or permuting of the five bits that encoded the characters.

Python tips

The carriage return is written "`\r`" in strings and bytes, and line feed is "`\n`" (think "return" and "new line"). The end of a line usually has both of these: `\r\n`. A null character is written "`\0`". The "bell" character is "`\a`", and if your terminal supports it, it will be converted to a sound (this author's terminal flashes a bell icon but makes no noise). It is possible that the "enquire" character is "`\x05`", but we will not need this character in this book.

Programming tasks

1. Implement an encoder/decoder that can translate between text containing letters, punctuation, and digits and five-bit Baudot.

2. Implement an encoder/decoder that can translate between five-bit Baudot and the Bletchley notation.

3. Create a function that takes text in Bletchley notation and prints it with spaces, punctuation, digits, and end-of-line characters.

4. Find a large piece of text, including spaces, punctuation, and occasional digits, and encode it with Bletchley Baudot. Use the result to compile a table of frequencies for the 32 symbols. Put this table into a data structure that you can import into your Python scripts.

bits	letter mode	symbol mode	Bletchley notation
00000	[null]	[null]	/
10000	E	3	E
01000	[carriage return]	[carriage return]	4
11000	A	-	A
00100	[space]	[space]	9
10100	S	'	S
01100	I	8	I
11100	U	7	U
00010	[line feed]	[line feed]	3
10010	D	[enquire]	D
01010	R	4	R
11010	J	[bell]	J
00110	N	,	N
10110	F	!	F
01110	C	:	C
11110	K	(	K
00001	T	5	T
10001	Z	+	Z
01001	L	)	L
11001	W	2	W
00101	H	£	H
10101	Y	6	Y
01101	P	0	P
11101	Q	1	Q
00011	O	9	O
10011	B	?	B
01011	G	&	G
11011	[symbol shift]		+
00111	M	.	M
10111	X	/	X
01111	V	;	V
11111		[letter shift]	8

Table 181.1: Baudot symbol table.