2010 Solutions

(E) Texting, Texting, One Two Three (1/2)

From examining repeated elements and letters, we can work out most, but not all, of the character codes for the letters, along with SPACE being 1, the SHIFT sequence that creates a capital letter being 33, and the END MESSAGE sequence being 331 (SHIFT + SPACE, a sequence that otherwise wouldn't be used).

Lowercase 'z' doesn't appear in the plaintext, but knowing that uppercase 'Z' is 3323444 and "shift" is 33 we can conclude that lowercase 'z' is 23444.

The system we find is a "variable-length", rather than "fixed-length", code system. Although some of the codes are much longer than three digits, overall most codes are much shorter, because very common characters (like e, t, "space", etc.) are given very short codes whereas only fairly rare letters are given the longer codes.

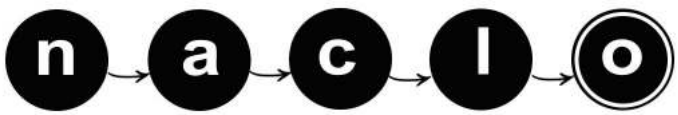
a	31	**n**	42
b	2341	**o**	32
c	242	**p**	342
d	233	**q**	23442
e	21	**r**	44
f	244	**s**	43
g	341	**t**	22
h	231	**u**	241
i	41	**v**	2342
j	23443	**w**	344
k	2343	**x**	23441
l	232	**y**	343
m	243	**z**	23444

Two letters remain, however, 'r' and 'x', neither of which appear in the plaintext. To determine their values, we have to work out the overall logic of the system.

Looking at the numerical codes, we notice that they aren't random: there are frequently repeated initial subcodes, and a lot of gaps. For example, many codes begin in 23-, 234-, and 34-, but none begin in, for example, 1-.

But why shouldn't a code begin with 1? If you consider the use of such a device, what would happen if a letter code began with one? What would happen is that, since 1 is "space", the device wouldn't know whether that 1 was intended as a space or as the first number of a longer code.

Looking further, we can see that *none* of the codes begins with another letter's code. That is, since 'a' is 31, no other letters' codes have 31- as their first two numbers, since 'b' is 2341, no other codes have these as their first four numbers, etc.

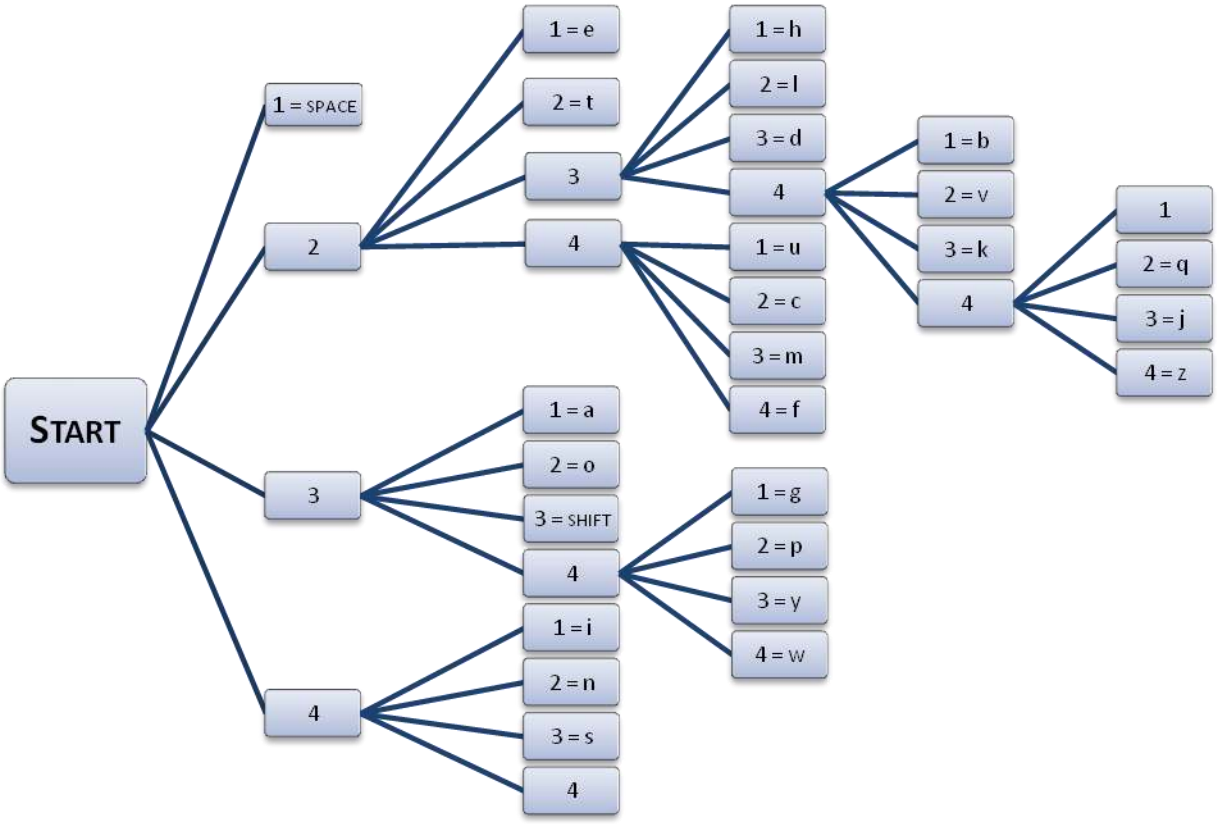


2010 Solutions

(E) Texting, Texting, One Two Three (2/2)

"Fixed-length" code systems, like the original three-number code system, always know when the user has keyed in a complete code. But since this system has "variable-length" codes, it needs some system to tell it whether some sequence, of whatever length, is a complete code or just the first part of a longer one. In this case, it knows when a code is complete because no beginning part of a valid code is a valid code.

It's especially clear if we draw a "tree" of the codes: only those nodes that don't have further "branches" are assigned characters. Assigning "31" to "a" is fine, because there aren't any "311", "312", etc. to confuse the system. On the other hand, we can't assign "34" to anything because then it would prevent "341", "342", etc. from being entered.



Looking carefully at our tree, there are exactly two "free" nodes – that is, ones that don't already have a character assigned and that don't have any "branches": "44" and "23441". These are where "r" and "x" have to go – if they go anywhere else, the internal logic of the system is compromised.

Since frequent letters (like "e","t","a","o","i","n","s") get short codes, and rare letters (like "q","j","z") get long codes, "r" must be "44" while "x" is "23441".

Now we have all 26 letters, SPACE, SHIFT, and the END sequence, and can encode and decode any message for this device.