# -\*- coding: utf-8 -\*-

# Copyright (c) 2013, Michael Nooner

# All rights reserved.

#

# Redistribution and use in source and binary forms, with or without

# modification, are permitted provided that the following conditions are met:

# \* Redistributions of source code must retain the above copyright

# notice, this list of conditions and the following disclaimer.

# \* Redistributions in binary form must reproduce the above copyright

# notice, this list of conditions and the following disclaimer in the

# documentation and/or other materials provided with the distribution.

# \* Neither the name of the copyright holder nor the names of its

# contributors may be used to endorse or promote products derived from

# this software without specific prior written permission

#

# THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"

# AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE

# IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE

# ARE DISCLAIMED. IN NO EVENT SHALL <COPYRIGHT HOLDER> BE LIABLE FOR ANY

# DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

# (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

# LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND

# ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

# (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS

# SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

"""This module lists out all of the tables needed to create a QR code.

If you are viewing this in the HTML documentation, I recommend reading the

actual file instead. The formating for the tables is much more readable.

"""

from \_\_future\_\_ import division, unicode\_literals

#: This defines the QR Code's 'mode' which sets what

#: type of code it is along with its size.

modes = {

'numeric': 1,

'alphanumeric': 2,

'binary': 4,

'kanji': 8,

}

#: This defines the amount of error correction. The dictionary

#: allows the user to specify this in several ways.

error\_level = {'L': 'L', 'l': 'L', '7%': 'L', .7: 'L',

'M': 'M', 'm': 'M', '15%': 'M', .15: 'M',

'Q': 'Q', 'q': 'Q', '25%': 'Q', .25: 'Q',

'H': 'H', 'h': 'H', '30%': 'H', .30: 'H'}

#: This is a dictionary holds how long the "data length" field is for

#: each version and mode of the QR Code.

data\_length\_field = {9: {1: 10, 2: 9, 4: 8, 8: 8},

26: {1: 12, 2: 11, 4: 16, 8: 10},

40: {1: 14, 2: 13, 4: 16, 8: 12}}

#: QR Codes uses a unique ASCII-like table for the 'alphanumeric' mode.

#: This is a dictionary representing that unique table, where the

#: keys are the possible characters in the data and the values

#: are the character's numeric representation.

ascii\_codes = {'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7,

'8': 8, '9': 9, 'A': 10, 'B': 11, 'C': 12, 'D': 13, 'E': 14,

'F': 15, 'G': 16, 'H': 17, 'I': 18, 'J': 19, 'K': 20, 'L': 21,

'M': 22, 'N': 23, 'O': 24, 'P': 25, 'Q': 26, 'R': 27, 'S': 28,

'T': 29, 'U': 30, 'V': 31, 'W': 32, 'X': 33, 'Y': 34, 'Z': 35,

' ': 36, '$': 37, '%': 38, '\*': 39, '+': 40, '-': 41, '.': 42,

'/': 43, ':': 44}

#: This array specifies the size of a QR Code in pixels. These numbers are

#: defined in the standard. The indexes correspond to the QR Code's

#: version number. This array was taken from:

#:

#: http://www.denso-wave.com/qrcode/vertable1-e.html

version\_size = [None, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57,

61, 65, 69, 73, 77, 81, 85, 89, 93, 97,

101, 105, 109, 113, 117, 121, 125, 129, 133, 137,

141, 145, 149, 153, 157, 161, 165, 169, 173, 177]

#: This dictionary lists the data capacity for all possible QR Codes.

#: This dictionary is organized where the first key corresponds to the

#: QR Code version number. The next key corresponds to the error

#: correction level, see error. The final key corresponds to

#: the mode number, see modes. The zero mode number represents the

#: possible "data bits." This table was taken from:

#:

#: http://www.denso-wave.com/qrcode/vertable1-e.html

data\_capacity = {

1: {

"L": {0: 152, 1: 41, 2: 25, 4: 17, 8: 10, },

"M": {0: 128, 1: 34, 2: 20, 4: 14, 8: 8, },

"Q": {0: 104, 1: 27, 2: 16, 4: 11, 8: 7, },

"H": {0: 72, 1: 17, 2: 10, 4: 7, 8: 4, }},

2: {

"L": {0: 272, 1: 77, 2: 47, 4: 32, 8: 20, },

"M": {0: 224, 1: 63, 2: 38, 4: 26, 8: 16, },

"Q": {0: 176, 1: 48, 2: 29, 4: 20, 8: 12, },

"H": {0: 128, 1: 34, 2: 20, 4: 14, 8: 8, }},

3: {

"L": {0: 440, 1: 127, 2: 77, 4: 53, 8: 32, },

"M": {0: 352, 1: 101, 2: 61, 4: 42, 8: 26, },

"Q": {0: 272, 1: 77, 2: 47, 4: 32, 8: 20, },

"H": {0: 208, 1: 58, 2: 35, 4: 24, 8: 15, }},

4: {

"L": {0: 640, 1: 187, 2: 114, 4: 78, 8: 48, },

"M": {0: 512, 1: 149, 2: 90, 4: 62, 8: 38, },

"Q": {0: 384, 1: 111, 2: 67, 4: 46, 8: 28, },

"H": {0: 288, 1: 82, 2: 50, 4: 34, 8: 21, }},

5: {

"L": {0: 864, 1: 255, 2: 154, 4: 106, 8: 65, },

"M": {0: 688, 1: 202, 2: 122, 4: 84, 8: 52, },

"Q": {0: 496, 1: 144, 2: 87, 4: 60, 8: 37, },

"H": {0: 368, 1: 106, 2: 64, 4: 44, 8: 27, }},

6: {

"L": {0: 1088, 1: 322, 2: 195, 4: 134, 8: 82, },

"M": {0: 864, 1: 255, 2: 154, 4: 106, 8: 65, },

"Q": {0: 608, 1: 178, 2: 108, 4: 74, 8: 45, },

"H": {0: 480, 1: 139, 2: 84, 4: 58, 8: 36, }},

7: {

"L": {0: 1248, 1: 370, 2: 224, 4: 154, 8: 95, },

"M": {0: 992, 1: 293, 2: 178, 4: 122, 8: 75, },

"Q": {0: 704, 1: 207, 2: 125, 4: 86, 8: 53, },

"H": {0: 528, 1: 154, 2: 93, 4: 64, 8: 39, }},

8: {

"L": {0: 1552, 1: 461, 2: 279, 4: 192, 8: 118, },

"M": {0: 1232, 1: 365, 2: 221, 4: 152, 8: 93, },

"Q": {0: 880, 1: 259, 2: 157, 4: 108, 8: 66, },

"H": {0: 688, 1: 202, 2: 122, 4: 84, 8: 52, }},

9: {

"L": {0: 1856, 1: 552, 2: 335, 4: 230, 8: 141, },

"M": {0: 1456, 1: 432, 2: 262, 4: 180, 8: 111, },

"Q": {0: 1056, 1: 312, 2: 189, 4: 130, 8: 80, },

"H": {0: 800, 1: 235, 2: 143, 4: 98, 8: 60, }},

10: {

"L": {0: 2192, 1: 652, 2: 395, 4: 271, 8: 167, },

"M": {0: 1728, 1: 513, 2: 311, 4: 213, 8: 131, },

"Q": {0: 1232, 1: 364, 2: 221, 4: 151, 8: 93, },

"H": {0: 976, 1: 288, 2: 174, 4: 119, 8: 74, }},

11: {

"L": {0: 2592, 1: 772, 2: 468, 4: 321, 8: 198, },

"M": {0: 2032, 1: 604, 2: 366, 4: 251, 8: 155, },

"Q": {0: 1440, 1: 427, 2: 259, 4: 177, 8: 109, },

"H": {0: 1120, 1: 331, 2: 200, 4: 137, 8: 85, }},

12: {

"L": {0: 2960, 1: 883, 2: 535, 4: 367, 8: 226, },

"M": {0: 2320, 1: 691, 2: 419, 4: 287, 8: 177, },

"Q": {0: 1648, 1: 489, 2: 296, 4: 203, 8: 125, },

"H": {0: 1264, 1: 374, 2: 227, 4: 155, 8: 96, }},

13: {

"L": {0: 3424, 1: 1022, 2: 619, 4: 425, 8: 262, },

"M": {0: 2672, 1: 796, 2: 483, 4: 331, 8: 204, },

"Q": {0: 1952, 1: 580, 2: 352, 4: 241, 8: 149, },

"H": {0: 1440, 1: 427, 2: 259, 4: 177, 8: 109, }},

14: {

"L": {0: 3688, 1: 1101, 2: 667, 4: 458, 8: 282, },

"M": {0: 2920, 1: 871, 2: 528, 4: 362, 8: 223, },

"Q": {0: 2088, 1: 621, 2: 376, 4: 258, 8: 159, },

"H": {0: 1576, 1: 468, 2: 283, 4: 194, 8: 120, }},

15: {

"L": {0: 4184, 1: 1250, 2: 758, 4: 520, 8: 320, },

"M": {0: 3320, 1: 991, 2: 600, 4: 412, 8: 254, },

"Q": {0: 2360, 1: 703, 2: 426, 4: 292, 8: 180, },

"H": {0: 1784, 1: 530, 2: 321, 4: 220, 8: 136, }},

16: {

"L": {0: 4712, 1: 1408, 2: 854, 4: 586, 8: 361, },

"M": {0: 3624, 1: 1082, 2: 656, 4: 450, 8: 277, },

"Q": {0: 2600, 1: 775, 2: 470, 4: 322, 8: 198, },

"H": {0: 2024, 1: 602, 2: 365, 4: 250, 8: 154, }},

17: {

"L": {0: 5176, 1: 1548, 2: 938, 4: 644, 8: 397, },

"M": {0: 4056, 1: 1212, 2: 734, 4: 504, 8: 310, },

"Q": {0: 2936, 1: 876, 2: 531, 4: 364, 8: 224, },

"H": {0: 2264, 1: 674, 2: 408, 4: 280, 8: 173, }},

18: {

"L": {0: 5768, 1: 1725, 2: 1046, 4: 718, 8: 442, },

"M": {0: 4504, 1: 1346, 2: 816, 4: 560, 8: 345, },

"Q": {0: 3176, 1: 948, 2: 574, 4: 394, 8: 243, },

"H": {0: 2504, 1: 746, 2: 452, 4: 310, 8: 191, }},

19: {

"L": {0: 6360, 1: 1903, 2: 1153, 4: 792, 8: 488, },

"M": {0: 5016, 1: 1500, 2: 909, 4: 624, 8: 384, },

"Q": {0: 3560, 1: 1063, 2: 644, 4: 442, 8: 272, },

"H": {0: 2728, 1: 813, 2: 493, 4: 338, 8: 208, }},

20: {

"L": {0: 6888, 1: 2061, 2: 1249, 4: 858, 8: 528, },

"M": {0: 5352, 1: 1600, 2: 970, 4: 666, 8: 410, },

"Q": {0: 3880, 1: 1159, 2: 702, 4: 482, 8: 297, },

"H": {0: 3080, 1: 919, 2: 557, 4: 382, 8: 235, }},

21: {

"L": {0: 7456, 1: 2232, 2: 1352, 4: 929, 8: 572, },

"M": {0: 5712, 1: 1708, 2: 1035, 4: 711, 8: 438, },

"Q": {0: 4096, 1: 1224, 2: 742, 4: 509, 8: 314, },

"H": {0: 3248, 1: 969, 2: 587, 4: 403, 8: 248, }},

22: {

"L": {0: 8048, 1: 2409, 2: 1460, 4: 1003, 8: 618, },

"M": {0: 6256, 1: 1872, 2: 1134, 4: 779, 8: 480, },

"Q": {0: 4544, 1: 1358, 2: 823, 4: 565, 8: 348, },

"H": {0: 3536, 1: 1056, 2: 640, 4: 439, 8: 270, }},

23: {

"L": {0: 8752, 1: 2620, 2: 1588, 4: 1091, 8: 672, },

"M": {0: 6880, 1: 2059, 2: 1248, 4: 857, 8: 528, },

"Q": {0: 4912, 1: 1468, 2: 890, 4: 611, 8: 376, },

"H": {0: 3712, 1: 1108, 2: 672, 4: 461, 8: 284, }},

24: {

"L": {0: 9392, 1: 2812, 2: 1704, 4: 1171, 8: 721, },

"M": {0: 7312, 1: 2188, 2: 1326, 4: 911, 8: 561, },

"Q": {0: 5312, 1: 1588, 2: 963, 4: 661, 8: 407, },

"H": {0: 4112, 1: 1228, 2: 744, 4: 511, 8: 315, }},

25: {

"L": {0: 10208, 1: 3057, 2: 1853, 4: 1273, 8: 784, },

"M": {0: 8000, 1: 2395, 2: 1451, 4: 997, 8: 614, },

"Q": {0: 5744, 1: 1718, 2: 1041, 4: 715, 8: 440, },

"H": {0: 4304, 1: 1286, 2: 779, 4: 535, 8: 330, }},

26: {

"L": {0: 10960, 1: 3283, 2: 1990, 4: 1367, 8: 842, },

"M": {0: 8496, 1: 2544, 2: 1542, 4: 1059, 8: 652, },

"Q": {0: 6032, 1: 1804, 2: 1094, 4: 751, 8: 462, },

"H": {0: 4768, 1: 1425, 2: 864, 4: 593, 8: 365, }},

27: {

"L": {0: 11744, 1: 3514, 2: 2132, 4: 1465, 8: 902, },

"M": {0: 9024, 1: 2701, 2: 1637, 4: 1125, 8: 692, },

"Q": {0: 6464, 1: 1933, 2: 1172, 4: 805, 8: 496, },

"H": {0: 5024, 1: 1501, 2: 910, 4: 625, 8: 385, }},

28: {

"L": {0: 12248, 1: 3669, 2: 2223, 4: 1528, 8: 940, },

"M": {0: 9544, 1: 2857, 2: 1732, 4: 1190, 8: 732, },

"Q": {0: 6968, 1: 2085, 2: 1263, 4: 868, 8: 534, },

"H": {0: 5288, 1: 1581, 2: 958, 4: 658, 8: 405, }},

29: {

"L": {0: 13048, 1: 3909, 2: 2369, 4: 1628, 8: 1002, },

"M": {0: 10136, 1: 3035, 2: 1839, 4: 1264, 8: 778, },

"Q": {0: 7288, 1: 2181, 2: 1322, 4: 908, 8: 559, },

"H": {0: 5608, 1: 1677, 2: 1016, 4: 698, 8: 430, }},

30: {

"L": {0: 13880, 1: 4158, 2: 2520, 4: 1732, 8: 1066, },

"M": {0: 10984, 1: 3289, 2: 1994, 4: 1370, 8: 843, },

"Q": {0: 7880, 1: 2358, 2: 1429, 4: 982, 8: 604, },

"H": {0: 5960, 1: 1782, 2: 1080, 4: 742, 8: 457, }},

31: {

"L": {0: 14744, 1: 4417, 2: 2677, 4: 1840, 8: 1132, },

"M": {0: 11640, 1: 3486, 2: 2113, 4: 1452, 8: 894, },

"Q": {0: 8264, 1: 2473, 2: 1499, 4: 1030, 8: 634, },

"H": {0: 6344, 1: 1897, 2: 1150, 4: 790, 8: 486, }},

32: {

"L": {0: 15640, 1: 4686, 2: 2840, 4: 1952, 8: 1201, },

"M": {0: 12328, 1: 3693, 2: 2238, 4: 1538, 8: 947, },

"Q": {0: 8920, 1: 2670, 2: 1618, 4: 1112, 8: 684, },

"H": {0: 6760, 1: 2022, 2: 1226, 4: 842, 8: 518, }},

33: {

"L": {0: 16568, 1: 4965, 2: 3009, 4: 2068, 8: 1273, },

"M": {0: 13048, 1: 3909, 2: 2369, 4: 1628, 8: 1002, },

"Q": {0: 9368, 1: 2805, 2: 1700, 4: 1168, 8: 719, },

"H": {0: 7208, 1: 2157, 2: 1307, 4: 898, 8: 553, }},

34: {

"L": {0: 17528, 1: 5253, 2: 3183, 4: 2188, 8: 1347, },

"M": {0: 13800, 1: 4134, 2: 2506, 4: 1722, 8: 1060, },

"Q": {0: 9848, 1: 2949, 2: 1787, 4: 1228, 8: 756, },

"H": {0: 7688, 1: 2301, 2: 1394, 4: 958, 8: 590, }},

35: {

"L": {0: 18448, 1: 5529, 2: 3351, 4: 2303, 8: 1417, },

"M": {0: 14496, 1: 4343, 2: 2632, 4: 1809, 8: 1113, },

"Q": {0: 10288, 1: 3081, 2: 1867, 4: 1283, 8: 790, },

"H": {0: 7888, 1: 2361, 2: 1431, 4: 983, 8: 605, }},

36: {

"L": {0: 19472, 1: 5836, 2: 3537, 4: 2431, 8: 1496, },

"M": {0: 15312, 1: 4588, 2: 2780, 4: 1911, 8: 1176, },

"Q": {0: 10832, 1: 3244, 2: 1966, 4: 1351, 8: 832, },

"H": {0: 8432, 1: 2524, 2: 1530, 4: 1051, 8: 647, }},

37: {

"L": {0: 20528, 1: 6153, 2: 3729, 4: 2563, 8: 1577, },

"M": {0: 15936, 1: 4775, 2: 2894, 4: 1989, 8: 1224, },

"Q": {0: 11408, 1: 3417, 2: 2071, 4: 1423, 8: 876, },

"H": {0: 8768, 1: 2625, 2: 1591, 4: 1093, 8: 673, }},

38: {

"L": {0: 21616, 1: 6479, 2: 3927, 4: 2699, 8: 1661, },

"M": {0: 16816, 1: 5039, 2: 3054, 4: 2099, 8: 1292, },

"Q": {0: 12016, 1: 3599, 2: 2181, 4: 1499, 8: 923, },

"H": {0: 9136, 1: 2735, 2: 1658, 4: 1139, 8: 701, }},

39: {

"L": {0: 22496, 1: 6743, 2: 4087, 4: 2809, 8: 1729, },

"M": {0: 17728, 1: 5313, 2: 3220, 4: 2213, 8: 1362, },

"Q": {0: 12656, 1: 3791, 2: 2298, 4: 1579, 8: 972, },

"H": {0: 9776, 1: 2927, 2: 1774, 4: 1219, 8: 750, }},

40: {

"L": {0: 23648, 1: 7089, 2: 4296, 4: 2953, 8: 1817, },

"M": {0: 18672, 1: 5596, 2: 3391, 4: 2331, 8: 1435, },

"Q": {0: 13328, 1: 3993, 2: 2420, 4: 1663, 8: 1024, },

"H": {0: 10208, 1: 3057, 2: 1852, 4: 1273, 8: 784, }}

}

#: This table defines the "Error Correction Code Words and Block Information."

#: The table lists the number of error correction words that are required

#: to be generated for each version and error correction level. The table

#: is accessed by first using the version number as a key and then the

#: error level. The array values correspond to these columns from the source

#: table:

#:

#: +----------------------------+

#: |0 | EC Code Words Per Block |

#: +----------------------------+

#: |1 | Block 1 Count |

#: +----------------------------+

#: |2 | Block 1 Data Code Words |

#: +----------------------------+

#: |3 | Block 2 Count |

#: +----------------------------+

#: |4 | Block 2 Data Code Words |

#: +----------------------------+

#:

#: This table was taken from:

#:

#: http://www.thonky.com/qr-code-tutorial/error-correction-table/

eccwbi = {

1: {

'L': [7, 1, 19, 0, 0, ],

'M': [10, 1, 16, 0, 0, ],

'Q': [13, 1, 13, 0, 0, ],

'H': [17, 1, 9, 0, 0, ],

},

2: {

'L': [10, 1, 34, 0, 0, ],

'M': [16, 1, 28, 0, 0, ],

'Q': [22, 1, 22, 0, 0, ],

'H': [28, 1, 16, 0, 0, ],

},

3: {

'L': [15, 1, 55, 0, 0, ],

'M': [26, 1, 44, 0, 0, ],

'Q': [18, 2, 17, 0, 0, ],

'H': [22, 2, 13, 0, 0, ],

},

4: {

'L': [20, 1, 80, 0, 0, ],

'M': [18, 2, 32, 0, 0, ],

'Q': [26, 2, 24, 0, 0, ],

'H': [16, 4, 9, 0, 0, ],

},

5: {

'L': [26, 1, 108, 0, 0, ],

'M': [24, 2, 43, 0, 0, ],

'Q': [18, 2, 15, 2, 16, ],

'H': [22, 2, 11, 2, 12, ],

},

6: {

'L': [18, 2, 68, 0, 0, ],

'M': [16, 4, 27, 0, 0, ],

'Q': [24, 4, 19, 0, 0, ],

'H': [28, 4, 15, 0, 0, ],

},

7: {

'L': [20, 2, 78, 0, 0, ],

'M': [18, 4, 31, 0, 0, ],

'Q': [18, 2, 14, 4, 15, ],

'H': [26, 4, 13, 1, 14, ],

},

8: {

'L': [24, 2, 97, 0, 0, ],

'M': [22, 2, 38, 2, 39, ],

'Q': [22, 4, 18, 2, 19, ],

'H': [26, 4, 14, 2, 15, ],

},

9: {

'L': [30, 2, 116, 0, 0, ],

'M': [22, 3, 36, 2, 37, ],

'Q': [20, 4, 16, 4, 17, ],

'H': [24, 4, 12, 4, 13, ],

},

10: {

'L': [18, 2, 68, 2, 69, ],

'M': [26, 4, 43, 1, 44, ],

'Q': [24, 6, 19, 2, 20, ],

'H': [28, 6, 15, 2, 16, ],

},

11: {

'L': [20, 4, 81, 0, 0, ],

'M': [30, 1, 50, 4, 51, ],

'Q': [28, 4, 22, 4, 23, ],

'H': [24, 3, 12, 8, 13, ],

},

12: {

'L': [24, 2, 92, 2, 93, ],

'M': [22, 6, 36, 2, 37, ],

'Q': [26, 4, 20, 6, 21, ],

'H': [28, 7, 14, 4, 15, ],

},

13: {

'L': [26, 4, 107, 0, 0, ],

'M': [22, 8, 37, 1, 38, ],

'Q': [24, 8, 20, 4, 21, ],

'H': [22, 12, 11, 4, 12, ],

},

14: {

'L': [30, 3, 115, 1, 116, ],

'M': [24, 4, 40, 5, 41, ],

'Q': [20, 11, 16, 5, 17, ],

'H': [24, 11, 12, 5, 13, ],

},

15: {

'L': [22, 5, 87, 1, 88, ],

'M': [24, 5, 41, 5, 42, ],

'Q': [30, 5, 24, 7, 25, ],

'H': [24, 11, 12, 7, 13, ],

},

16: {

'L': [24, 5, 98, 1, 99, ],

'M': [28, 7, 45, 3, 46, ],

'Q': [24, 15, 19, 2, 20, ],

'H': [30, 3, 15, 13, 16, ],

},

17: {

'L': [28, 1, 107, 5, 108, ],

'M': [28, 10, 46, 1, 47, ],

'Q': [28, 1, 22, 15, 23, ],

'H': [28, 2, 14, 17, 15, ],

},

18: {

'L': [30, 5, 120, 1, 121, ],

'M': [26, 9, 43, 4, 44, ],

'Q': [28, 17, 22, 1, 23, ],

'H': [28, 2, 14, 19, 15, ],

},

19: {

'L': [28, 3, 113, 4, 114, ],

'M': [26, 3, 44, 11, 45, ],

'Q': [26, 17, 21, 4, 22, ],

'H': [26, 9, 13, 16, 14, ],

},

20: {

'L': [28, 3, 107, 5, 108, ],

'M': [26, 3, 41, 13, 42, ],

'Q': [30, 15, 24, 5, 25, ],

'H': [28, 15, 15, 10, 16, ],

},

21: {

'L': [28, 4, 116, 4, 117, ],

'M': [26, 17, 42, 0, 0, ],

'Q': [28, 17, 22, 6, 23, ],

'H': [30, 19, 16, 6, 17, ],

},

22: {

'L': [28, 2, 111, 7, 112, ],

'M': [28, 17, 46, 0, 0, ],

'Q': [30, 7, 24, 16, 25, ],

'H': [24, 34, 13, 0, 0, ],

},

23: {

'L': [30, 4, 121, 5, 122, ],

'M': [28, 4, 47, 14, 48, ],

'Q': [30, 11, 24, 14, 25, ],

'H': [30, 16, 15, 14, 16, ],

},

24: {

'L': [30, 6, 117, 4, 118, ],

'M': [28, 6, 45, 14, 46, ],

'Q': [30, 11, 24, 16, 25, ],

'H': [30, 30, 16, 2, 17, ],

},

25: {

'L': [26, 8, 106, 4, 107, ],

'M': [28, 8, 47, 13, 48, ],

'Q': [30, 7, 24, 22, 25, ],

'H': [30, 22, 15, 13, 16, ],

},

26: {

'L': [28, 10, 114, 2, 115, ],

'M': [28, 19, 46, 4, 47, ],

'Q': [28, 28, 22, 6, 23, ],

'H': [30, 33, 16, 4, 17, ],

},

27: {

'L': [30, 8, 122, 4, 123, ],

'M': [28, 22, 45, 3, 46, ],

'Q': [30, 8, 23, 26, 24, ],

'H': [30, 12, 15, 28, 16, ],

},

28: {

'L': [30, 3, 117, 10, 118, ],

'M': [28, 3, 45, 23, 46, ],

'Q': [30, 4, 24, 31, 25, ],

'H': [30, 11, 15, 31, 16, ],

},

29: {

'L': [30, 7, 116, 7, 117, ],

'M': [28, 21, 45, 7, 46, ],

'Q': [30, 1, 23, 37, 24, ],

'H': [30, 19, 15, 26, 16, ],

},

30: {

'L': [30, 5, 115, 10, 116, ],

'M': [28, 19, 47, 10, 48, ],

'Q': [30, 15, 24, 25, 25, ],

'H': [30, 23, 15, 25, 16, ],

},

31: {

'L': [30, 13, 115, 3, 116, ],

'M': [28, 2, 46, 29, 47, ],

'Q': [30, 42, 24, 1, 25, ],

'H': [30, 23, 15, 28, 16, ],

},

32: {

'L': [30, 17, 115, 0, 0, ],

'M': [28, 10, 46, 23, 47, ],

'Q': [30, 10, 24, 35, 25, ],

'H': [30, 19, 15, 35, 16, ],

},

33: {

'L': [30, 17, 115, 1, 116, ],

'M': [28, 14, 46, 21, 47, ],

'Q': [30, 29, 24, 19, 25, ],

'H': [30, 11, 15, 46, 16, ],

},

34: {

'L': [30, 13, 115, 6, 116, ],

'M': [28, 14, 46, 23, 47, ],

'Q': [30, 44, 24, 7, 25, ],

'H': [30, 59, 16, 1, 17, ],

},

35: {

'L': [30, 12, 121, 7, 122, ],

'M': [28, 12, 47, 26, 48, ],

'Q': [30, 39, 24, 14, 25, ],

'H': [30, 22, 15, 41, 16, ],

},

36: {

'L': [30, 6, 121, 14, 122, ],

'M': [28, 6, 47, 34, 48, ],

'Q': [30, 46, 24, 10, 25, ],

'H': [30, 2, 15, 64, 16, ],

},

37: {

'L': [30, 17, 122, 4, 123, ],

'M': [28, 29, 46, 14, 47, ],

'Q': [30, 49, 24, 10, 25, ],

'H': [30, 24, 15, 46, 16, ],

},

38: {

'L': [30, 4, 122, 18, 123, ],

'M': [28, 13, 46, 32, 47, ],

'Q': [30, 48, 24, 14, 25, ],

'H': [30, 42, 15, 32, 16, ],

},

39: {

'L': [30, 20, 117, 4, 118, ],

'M': [28, 40, 47, 7, 48, ],

'Q': [30, 43, 24, 22, 25, ],

'H': [30, 10, 15, 67, 16, ],

},

40: {

'L': [30, 19, 118, 6, 119, ],

'M': [28, 18, 47, 31, 48, ],

'Q': [30, 34, 24, 34, 25, ],

'H': [30, 20, 15, 61, 16, ],

},

}

#: This table lists all of the generator polynomials used by QR Codes.

#: They are indexed by the number of "ECC Code Words" (see table above).

#: This table is taken from:

#:

#: http://www.matchadesign.com/blog/qr-code-demystified-part-4/

generator\_polynomials = {

7: [87, 229, 146, 149, 238, 102, 21],

10: [251, 67, 46, 61, 118, 70, 64, 94, 32, 45],

13: [74, 152, 176, 100, 86, 100, 106, 104, 130, 218, 206, 140, 78],

15: [8, 183, 61, 91, 202, 37, 51, 58, 58, 237, 140, 124, 5, 99, 105],

16: [120, 104, 107, 109, 102, 161, 76, 3, 91, 191, 147, 169, 182, 194,

225, 120],

17: [43, 139, 206, 78, 43, 239, 123, 206, 214, 147, 24, 99, 150, 39,

243, 163, 136],

18: [215, 234, 158, 94, 184, 97, 118, 170, 79, 187, 152, 148, 252, 179,

5, 98, 96, 153],

20: [17, 60, 79, 50, 61, 163, 26, 187, 202, 180, 221, 225, 83, 239, 156,

164, 212, 212, 188, 190],

22: [210, 171, 247, 242, 93, 230, 14, 109, 221, 53, 200, 74, 8, 172, 98,

80, 219, 134, 160, 105, 165, 231],

24: [229, 121, 135, 48, 211, 117, 251, 126, 159, 180, 169, 152, 192, 226,

228, 218, 111, 0, 117, 232, 87, 96, 227, 21],

26: [173, 125, 158, 2, 103, 182, 118, 17, 145, 201, 111, 28, 165, 53, 161,

21, 245, 142, 13, 102, 48, 227, 153, 145, 218, 70],

28: [168, 223, 200, 104, 224, 234, 108, 180, 110, 190, 195, 147, 205, 27,

232, 201, 21, 43, 245, 87, 42, 195, 212, 119, 242, 37, 9, 123],

30: [41, 173, 145, 152, 216, 31, 179, 182, 50, 48, 110, 86, 239, 96, 222,

125, 42, 173, 226, 193, 224, 130, 156, 37, 251, 216, 238, 40, 192,

180]

}

#: This table contains the log and values used in GF(256) arithmetic.

#: They are used to generate error correction codes for QR Codes.

#: This table is taken from:

#:

#: vhttp://www.thonky.com/qr-code-tutorial/log-antilog-table/

galois\_log = [

1, 2, 4, 8, 16, 32, 64, 128, 29, 58, 116, 232, 205, 135, 19, 38, 76, 152,

45, 90, 180, 117, 234, 201, 143, 3, 6, 12, 24, 48, 96, 192, 157, 39, 78,

156, 37, 74, 148, 53, 106, 212, 181, 119, 238, 193, 159, 35, 70, 140, 5,

10, 20, 40, 80, 160, 93, 186, 105, 210, 185, 111, 222, 161, 95, 190, 97,

194, 153, 47, 94, 188, 101, 202, 137, 15, 30, 60, 120, 240, 253, 231, 211,

187, 107, 214, 177, 127, 254, 225, 223, 163, 91, 182, 113, 226, 217, 175,

67, 134, 17, 34, 68, 136, 13, 26, 52, 104, 208, 189, 103, 206, 129, 31,

62, 124, 248, 237, 199, 147, 59, 118, 236, 197, 151, 51, 102, 204, 133,

23, 46, 92, 184, 109, 218, 169, 79, 158, 33, 66, 132, 21, 42, 84, 168, 77,

154, 41, 82, 164, 85, 170, 73, 146, 57, 114, 228, 213, 183, 115, 230, 209,

191, 99, 198, 145, 63, 126, 252, 229, 215, 179, 123, 246, 241, 255, 227,

219, 171, 75, 150, 49, 98, 196, 149, 55, 110, 220, 165, 87, 174, 65, 130,

25, 50, 100, 200, 141, 7, 14, 28, 56, 112, 224, 221, 167, 83, 166, 81,

162, 89, 178, 121, 242, 249, 239, 195, 155, 43, 86, 172, 69, 138, 9, 18,

36, 72, 144, 61, 122, 244, 245, 247, 243, 251, 235, 203, 139, 11, 22, 44,

88, 176, 125, 250, 233, 207, 131, 27, 54, 108, 216, 173, 71, 142, 1,]

#: This table contains the antilog and values used in GF(256) arithmetic.

#: They are used to generate error correction codes for QR Codes.

#: This table is taken from:

#:

#: http://www.thonky.com/qr-code-tutorial/log-antilog-table/

galois\_antilog = [

None, 0, 1, 25, 2, 50, 26, 198, 3, 223, 51, 238, 27, 104, 199, 75, 4, 100,

224, 14, 52, 141, 239, 129, 28, 193, 105, 248, 200, 8, 76, 113, 5, 138,

101, 47, 225, 36, 15, 33, 53, 147, 142, 218, 240, 18, 130, 69, 29, 181,

194, 125, 106, 39, 249, 185, 201, 154, 9, 120, 77, 228, 114, 166, 6, 191,

139, 98, 102, 221, 48, 253, 226, 152, 37, 179, 16, 145, 34, 136, 54, 208,

148, 206, 143, 150, 219, 189, 241, 210, 19, 92, 131, 56, 70, 64, 30, 66,

182, 163, 195, 72, 126, 110, 107, 58, 40, 84, 250, 133, 186, 61, 202, 94,

155, 159, 10, 21, 121, 43, 78, 212, 229, 172, 115, 243, 167, 87, 7, 112,

192, 247, 140, 128, 99, 13, 103, 74, 222, 237, 49, 197, 254, 24, 227, 165,

153, 119, 38, 184, 180, 124, 17, 68, 146, 217, 35, 32, 137, 46, 55, 63,

209, 91, 149, 188, 207, 205, 144, 135, 151, 178, 220, 252, 190, 97, 242,

86, 211, 171, 20, 42, 93, 158, 132, 60, 57, 83, 71, 109, 65, 162, 31, 45,

67, 216, 183, 123, 164, 118, 196, 23, 73, 236, 127, 12, 111, 246, 108,

161, 59, 82, 41, 157, 85, 170, 251, 96, 134, 177, 187, 204, 62, 90, 203,

89, 95, 176, 156, 169, 160, 81, 11, 245, 22, 235, 122, 117, 44, 215, 79,

174, 213, 233, 230, 231, 173, 232, 116, 214, 244, 234, 168, 80, 88, 175,]

#: This table contains the coordinates for the position adjustment patterns.

#: The index of the table corresponds to the QR Code's version number.

#: This table is taken from:

#:

#: http://www.thonky.com/qr-code-tutorial/part-3-mask-pattern/

position\_adjustment = [

None, #There is not version 0

None, #Version 1 does not need adjustment

[6, 18, ],

[6, 22, ],

[6, 26, ],

[6, 30, ],

[6, 34, ],

[6, 22, 38, ],

[6, 24, 42, ],

[6, 26, 46, ],

[6, 28, 50, ],

[6, 30, 54, ],

[6, 32, 58, ],

[6, 34, 62, ],

[6, 26, 46, 66, ],

[6, 26, 48, 70, ],

[6, 26, 50, 74, ],

[6, 30, 54, 78, ],

[6, 30, 56, 82, ],

[6, 30, 58, 86, ],

[6, 34, 62, 90, ],

[6, 28, 50, 72, 94, ],

[6, 26, 50, 74, 98, ],

[6, 30, 54, 78, 102, ],

[6, 28, 54, 80, 106, ],

[6, 32, 58, 84, 110, ],

[6, 30, 58, 86, 114, ],

[6, 34, 62, 90, 118, ],

[6, 26, 50, 74, 98, 122, ],

[6, 30, 54, 78, 102, 126, ],

[6, 26, 52, 78, 104, 130, ],

[6, 30, 56, 82, 108, 134, ],

[6, 34, 60, 86, 112, 138, ],

[6, 30, 58, 86, 114, 142, ],

[6, 34, 62, 90, 118, 146, ],

[6, 30, 54, 78, 102, 126, 150, ],

[6, 24, 50, 76, 102, 128, 154, ],

[6, 28, 54, 80, 106, 132, 158, ],

[6, 32, 58, 84, 110, 136, 162, ],

[6, 26, 54, 82, 110, 138, 166, ],

[6, 30, 58, 86, 114, 142, 170, ],

]

#: This table specifies the bit pattern to be added to a QR Code's

#: image to specify what version the code is. Note, this pattern

#: is not used for versions 1-6. This table is taken from:

#:

#: http://www.thonky.com/qr-code-tutorial/part-3-mask-pattern/

version\_pattern = [None, None, None, None, None, None, None, #0-6

'000111110010010100', '001000010110111100', '001001101010011001',

'001010010011010011', '001011101111110110', '001100011101100010',

'001101100001000111', '001110011000001101', '001111100100101000',

'010000101101111000', '010001010001011101', '010010101000010111',

'010011010100110010', '010100100110100110', '010101011010000011',

'010110100011001001', '010111011111101100', '011000111011000100',

'011001000111100001', '011010111110101011', '011011000010001110',

'011100110000011010', '011101001100111111', '011110110101110101',

'011111001001010000', '100000100111010101', '100001011011110000',

'100010100010111010', '100011011110011111', '100100101100001011',

'100101010000101110', '100110101001100100', '100111010101000001',

'101000110001101001'

]

#: This table contains the bit fields needed to specify the error code level and

#: mask pattern used by a QR Code. This table is take from:

#:

#: http://www.thonky.com/qr-code-tutorial/part-3-mask-pattern/

type\_bits = {

'L': {

0: '111011111000100',

1: '111001011110011',

2: '111110110101010',

3: '111100010011101',

4: '110011000101111',

5: '110001100011000',

6: '110110001000001',

7: '110100101110110',

},

'M': {

0: '101010000010010',

1: '101000100100101',

2: '101111001111100',

3: '101101101001011',

4: '100010111111001',

5: '100000011001110',

6: '100111110010111',

7: '100101010100000',

},

'Q': {

0: '011010101011111',

1: '011000001101000',

2: '011111100110001',

3: '011101000000110',

4: '010010010110100',

5: '010000110000011',

6: '010111011011010',

7: '010101111101101',

},

'H': {

0: '001011010001001',

1: '001001110111110',

2: '001110011100111',

3: '001100111010000',

4: '000011101100010',

5: '000001001010101',

6: '000110100001100',

7: '000100000111011',

},

}

#: This table contains \*functions\* to compute whether to change current bit when

#: creating the masks. All of the functions in the table return a boolean value.

#: A True result means you should add the bit to the QR Code exactly as is. A

#: False result means you should add the opposite bit. This table was taken

#: from:

#:

#: http://www.thonky.com/qr-code-tutorial/mask-patterns/

mask\_patterns = [

lambda row, col: (row + col) % 2 == 0,

lambda row, col: row % 2 == 0,

lambda row, col: col % 3 == 0,

lambda row, col: (row + col) % 3 == 0,

lambda row, col: ((row // 2) + (col // 3)) % 2 == 0,

lambda row, col: ((row \* col) % 2) + ((row \* col) % 3) == 0,

lambda row, col: (((row \* col) % 2) + ((row \* col) % 3)) % 2 == 0,

lambda row, col: (((row + col) % 2) + ((row \* col) % 3)) % 2 == 0]

#: This is a table of ASCII excape code for terminal colors. QR codes

#: are drawn using a space with a colored background. Hence, only

#: codes affecting background colors have been added.

#: http://misc.flogisoft.com/bash/tip\_colors\_and\_formatting

term\_colors = {

'default': 49,

'background': 49,

'reverse': 7,

'reversed': 7,

'inverse': 7,

'inverted': 7,

'black': 40,

'red': 41,

'green': 42,

'yellow': 43,

'blue': 44,

'magenta': 45,

'cyan': 46,

'light gray': 47,

'light grey': 47,

'dark gray': 100,

'dark grey': 100,

'light red': 101,

'light green': 102,

'light blue': 103,

'light yellow': 104,

'light magenta': 105,

'light cyan': 106,

'white': 107

}