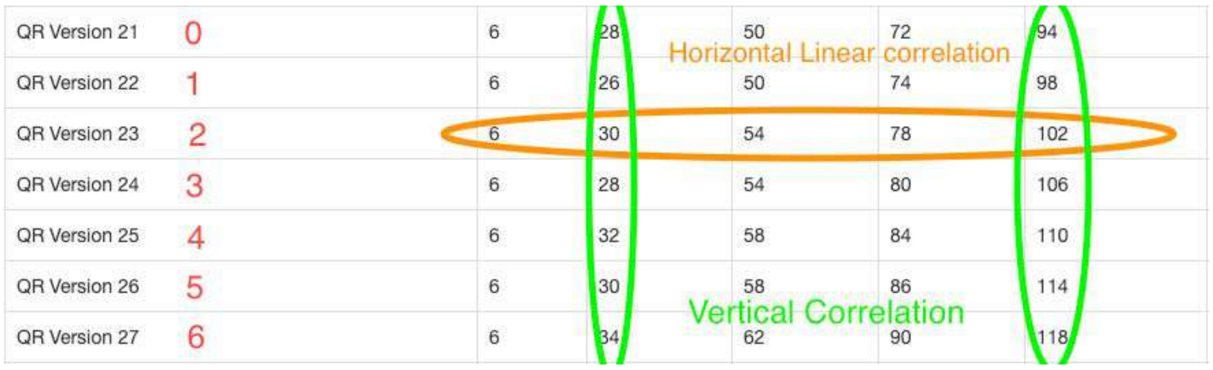
BONUS INFORMATION

1. Find best mask:
returns the best mask according to the penalty :
We evaluate the penalties of each mask (as described in the project description, without adding a white border), then choose the one with the least penalties.

2. Alignment patterns of all versions :

We avoided hard coding an array all the alignment patterns of all versions
instead we only work on the given version.
First we regrouped all the versions in groups of 7 based on the number of coordinates for the alignment patterns (1-6 ,7-13,14-20 .....)
Next , the given version is indexed based on its position in its group (ex : version 7 receives 0 , version 15 receives 1)
We found a vertical correlation for each group of 7 in the first and last column, and used them to find the other columns using the horizontal linear correlation, as shown below:



QR Version 21	0	6	28	50	72	94
QR Version 22	1	6	26	50	74	98
QR Version 23	2	6	30	54	78	102
QR Version 24	3	6	28	54	80	106
QR Version 25	4	6	32	58	84	110
QR Version 26	5	6	30	58	86	114
QR Version 27	6	6	34	62	90	118

Horizontal Linear correlation
Vertical Correlation

This function is coded in the “Extensions.java” and we implemented another function for the “addAlignmentPatterns” with the name “addAlignmentPatternsBonus”

Remark: We completed the version code words and error correction codewords arrays with the missing information in order to generate for versions greater than 4.
The QR codes(with versions greater than 5) that are generated cannot be read.

3. Choose best version

This function is coded in the “Extensions.java” and is used in the “main.java”.
This method gives the best version according to input’s length.