From my perspective, my code can only be used for exceptional cases. However, as the saying goes, the code that can solve the problem is the best code. So, I will use the most popular programming language (Python) to solve this problem.

I will solve a simple problem (the first question: L=5) first, so as to extract the key method of this coloring problem. After elaborating on the key method, I will apply this key method to complex problems (the second question: L=64).

1. Solve: L=5 with 12 red beads (R) and 13 blue beads (B)

I found that the first question can be imagined as a mini version of the chessboard, which is interlaced with black and white, respectively.

Put this idea in the first question. 13 staggered grids are starting from the top left corner in the 5 x 5 grid, precisely corresponding to the number of blue beads. These blue beads are staggered, so it is impossible for them to have the same color next to each other.

Meanwhile, just because the blue beads separate all the remaining grids, red beads can be directly put onto the remaining 12 grids. And it is also impossible for red beads to have the same color next to each other.

Therefore, according to the above statement, we only need to find staggered coordinates in the 5 x 5 grid, then we can solve the first question. Finding staggered grids is uncomplicated. It only needs to limit the addition of two coordinates is even or odd.

Using the code to implement, we can get the following code

For output data, please kindly find the attached file " output_question_5_1.txt ".
For code details, please kindly find the attached file " Q5_1.py " (Open with Python, Done by Pycharm).
For the convenience of review, I have copied all the code of the document as follows.

```python
l = 5    # The following sentence will be used

a = [[0 for i in range(l)] for i in range(l)]
# Create an empty grid without any beads

for i in range(l):
    for j in range(l):
        # The method to find the staggered coordinates
        if (i + j) % 2 != 0:
            a[i][j] = 'R'
        else:
            a[i][j] = 'B'
```

```
f = open("output_question_5_1.txt", "w+")    # Control the output

for i in range(l):
        s = ''
        for j in range(l):
                s = s + (a[i][j]) + ' '
        f.write(s)
        f.write('\n')
        # Write the results into the document

f.close()    # Finish the control
```

2. Method for Complex Problems

From the first section, the key method of coloring actually is very simple.

a) The first step is finding staggered grids by limiting the addition of two coordinates is even or odd.

b) The second step is finding approximately the same number of beads to fill these staggered grids.

c) The final step is putting the remaining beads onto the remaining grids.

3. Solve: L=64 with 139 red beads (R), 1451 blue beads (B), 977 green beads (G), 1072 white beads (W), 457 yellow beads (Y)

Luckily, I found that the number of staggered grids in the 64 x 64 grid is 2048, almost corresponding to the sum of green beads and white beads. Unfortunately, the sum of green beads and white beads is 2049, one more than staggered grids.

Therefore, The above key method needs to be changed a little. The modified method is to keep one green bead until the end. The remaining green and white beads will be put onto the staggered grids starting from the top left corner in the 64 x 64 grid.

We need to pay attention to here is that the only remaining green bead will be put at the bottom right corner. However, If I put white beads onto the staggered grid first and then green beads, there will be green beads around the vacancy near the bottom right corner in the end. Under these circumstances, the last green bead can not be put. So, I had to put green beads onto the staggered grid first so as to use up the green beads as soon as possible. On this condition, there will not be green beads around the vacancy near the bottom right corner in the end, but white beads around the vacancy.

The next step is putting red beads, blue beads, and yellow beads onto the remaining grids (The order doesn't matter here). Finally, don't forget to put the remaining green bead onto the last remaining space.

Using the code to implement, we can get the following code

For output data, please kindly find the attached file " output_question_5_2.txt ".
For code details, please kindly find the attached file " Q5_2.py " (Open with Python, Done by Pycharm).
For the convenience of review, I have copied all the code of the document as follows.

```
B = 1451
W = 1072
G = 977
Y = 457
R = 139
L = 64
# The following sentence will be used

A = [[0 for i in range(L)] for i in range(L)]
# Create an empty grid without any beads

for i in range(L):
    for j in range(L):
        # The method to find the staggered coordinates
        if (i + j) % 2 == 0:
            if G != 1:
                # Keep one green bead until the end
                # Put green beads onto the staggered grid FIRST
                A[i][j] = 'G'
                G = G - 1
            elif W != 0:
                A[i][j] = 'W'
                W = W - 1
                # After using up the green beads
                # THEN put white beads onto the staggered grid
        else:
            # Use up the blue beads, yellow beads, and red beads
            # The order doesn't matter here
            if B != 0:
                A[i][j] = 'B'
                B = B - 1
            elif Y != 0:
                A[i][j] = 'Y'
                Y = Y - 1
            elif R != 0:
                A[i][j] = 'R'
                R = R - 1

A[63][62] = 'G'
# Put the remaining green bead onto the last remaining space near the bottom right corner
```

```
f = open("output_question_5_2.txt", "w+")    # Control the output

for i in range(L):
        s = ''
        for j in range(L):
                s = s + (A[i][j]) + ' '
        f.write(s)
        f.write('\n')
        # Write the results into the document

f.close()    # Finish the control
```