We will create a frequency array that will store how many times each colour exists in the original string, excluding those times when there was a perfect hit.

**Below is the detailed algorithm:**

1. Make a helper function **slotScoreHelper(c)**, where ‘C’ is the color, it will return an integer depending on the current slot color.
   * If the current character is ‘B’, return 0.
   * If the current character is ‘G’, return 1.
   * If the current character is ‘R’, return 2.
   * If the current character is ‘Y’, return 3.
2. Make a frequency array **'COUNT'** to store each character’s occurrences in the original string and an integer **'POINTS'**to store the points scored.
3. Run a loop from(loop variable ‘i’) from 0 till 4 to calculate the perfect hits.
   * If 'ORIGINAL[i]' = 'GUESS[i]', we increment **points** by two since it’s a perfect hit.
   * Else, we make an integer variable **'COLOUR'**that will hold a value depending on the original string’s current character, i.e 'COLOUR' = slotScoreHelper('ORIGINAL[i]').
   * Increment 'COUNT[COLOUR]' by 1 to update the count of this current 'COLOUR' in the frequency array **'COUNT'**.
4. Run a loop from(loop variable ‘i’) from 0 till 4 to calculate the pseudo hits.
   * Similar to step 2(b), make an integer variable **'COLOUR',**which will hold a value depending on the current character of the guess string, i.e., 'COLOUR' = slotScoreHelepr('GUESS[i]').
   * If the current 'COLOUR' of our guess doesn’t matches with the current 'COLOUR' of the original, and 'COUNT[COLOUR]' > 0, i.e., the occurrence of this 'COLOUR' is greater than 0, then increment **'POINTS'**by one and decrement 'COUNT[COLOUR]' by 1 to mark that this 'COLOUR' has been considered once.
5. Return 'POINTS'**.**