Para começar guardamos o valor das peças que são possíveis de utilizar num array "bricks", estes valores são os únicos aceitados e por isso é um valor estático.

Ainda na leitura dos dados criou-se uma matriz “board” para criarmos um tabuleiro com as dimensões lidas na primeira linha do “input”. (ou um tabuleiro igual ao do input).

Para realizar a nossa analise de dados criamos o método “possibleCombinations” que vai calcular o numero de combinações possíveis a realizar no tabuleiro dado com as peças existentes guardadas neste caso no array “bricks”. Este método aceita como argumento (?a instancia/objecto/variável?) “sequenceSize” que vai conter o tamanho da sequencia que esta a ser analisada. Este método devolve “numOfCombinations” que contem o numero de combinações possíveis numa dada sequencia.

Neste método Inicializa-se um array (fala-se do tipo long?) “numOfCombinations” de tamanho “sequenceSize+1” isto para podermos analisar todos os valores da sequencia, no caso de criarmos apenas do tamanho de “sequenceSize“ quando fossemos a analisar o ultimo valor da sequencia daria erro!!(Barradas HELP!!) e declaramos o nosso caso base como “1” pois sem nenhuma peça só existe uma combinação possível ?e esta combinação sim seria 0?.

De seguida para o tamanho da sequencia analisada naquele momento vamos analisar para cada peça em “bricks” se a mesma é menor ou igual á sequencia para sabermos se a peça pode ou não ser utilizada para a sequencia em questão, no caso desta ser possível vamos adicionar ao “numOfCombinations” o valor dado tamanho da sequencia percorrido ate aquele ponto menos o valor da peça.

Na analise de dados vamos por começar por guardar na var “atPos” a posição “[i]” analisada e igualamos a currenColor para mais tarde podermos avaliar se existe uma sequencia. Se no primeiro caso a posição analisada não for um ponto “.” Igualamos o tamanho da nossa sequencia a 1 pois terá de ser uma cor. Continuando a analisar o tabuleiro a próxima posição vai ser guardada em “atPos” e se for um ponto ”.” e se a posição anterior (currentColor) for diferente de “atPos” isto é da posição que está a ser analisada, multiplica-se o valor de “result” pelo resultado devolvido pelo método “possibleCombinations(sequenceSize)” e reinicia-se o valor da sequencia a 0 pois neste caso a sequencia terminou.

No caso de estarmos a analisar o último valor do nosso tabuleiro “board” e o penúltimo valor analisado forem iguais multiplicamos “result” ao valor devolvido por “possibleCombinations” mas como argumento adicionamos 1 a “sequenceSize” pois a ultima posição ainda pertece á sequencia, no caso dos valores da ultima e penúltima posição não forem iguais vai multiplicar “result” pelo resultado de “possibleCombinations” com o tamanho da sequencia real que neste caso o valor devolvido sera 1.

No caso de não ser a ultima posição do tabuleiro e o valor lido fizer parte da sequencia ou seja “atPos”(vlaor lido) for igual a “currentColor”(valor anterior) adicionamos um á sequencia, se os dois valores não forem iguais multiplica-se ”result” pelo valor devolvido pelo nosso método e reinicia-se uma nova sequencia igualando os dois valores e a sequencia a 1.