MSc Statistical Programming assessed practical

P328

December 15, 2023

Rotten tomatoes

1. Throughout this question, data from the Rotten tomatoes movie rankings website, which contains movie ratings data, will be analysed to study various aspects of this dataset. A movie is categorised as fresh if at least 60 percent of the reviews are positive. By contrast, movies are classed as rotten if less than 60 percent of the reviews are positive. First, the percentage of movies rated fresh per year are shown from the year 2000 onwards (Fig.1). An upward trend in amount of movies that were rated fresh is shown. Given that, movies have remained the same quality since 2000 these data suggest that movie critics are being more generous in their overall ratings of movies since 2000. The graph entitled, percent of movies rated fresh per month (Fig.2) shows an upward trend in the percent of movies rated fresh as the months progress throughout the year. The month with the highest overall rating was November with over 67 percent of movies rated fresh, closely followed by December with over 66 percent of the movies rated fresh. By contrast, both January and February had less than 62 percent of movies rated fresh. The data imply that movie critics tend to be more generous as the year progresses. Finally, examination of the percent of movies rated fresh throughout the month (Fig.3) ranges from a 65 percent maximum average rating to a minimum rating of around 60 percent Although, the 60 percent, seems fairly low compared with the next lowest value of over 62.5 percent. No obvious trend is observed in the plot throughout the month.

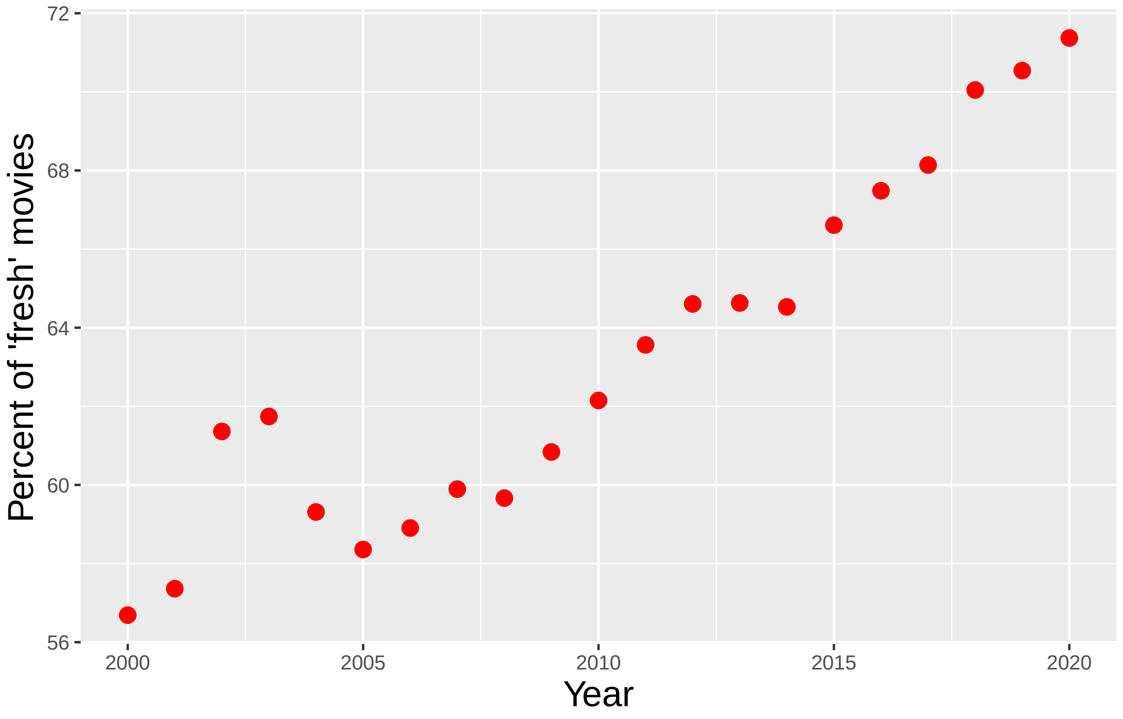


Figure 1: Percent of movies that are rated fresh per year

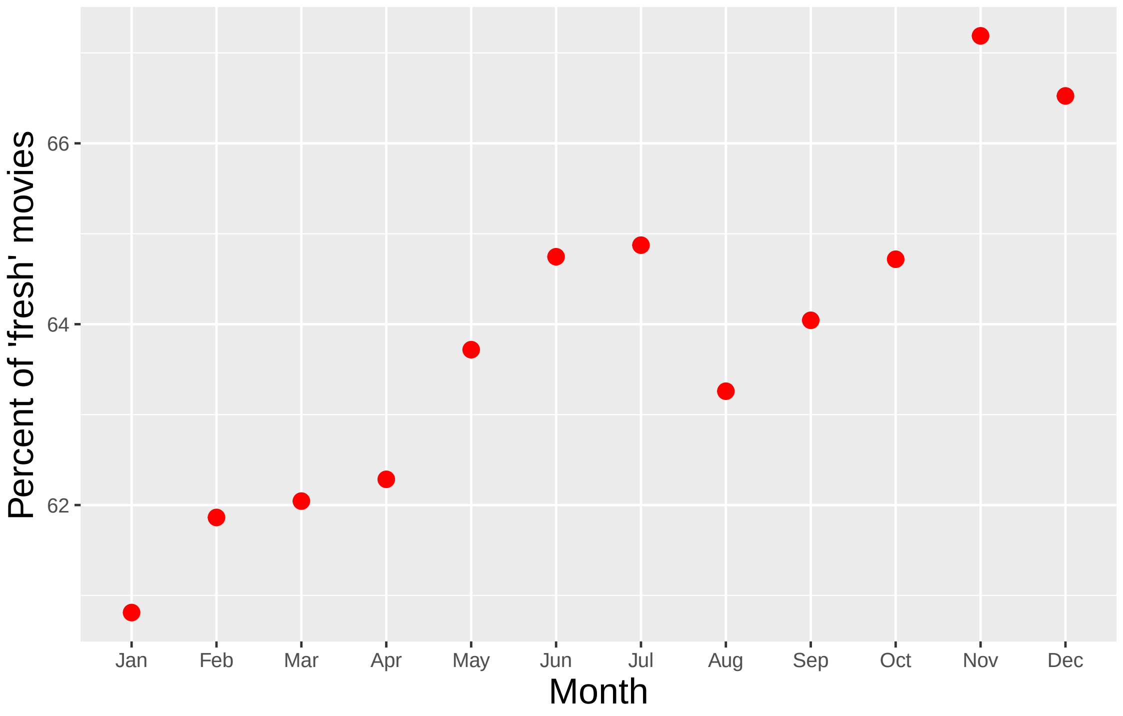


Figure 2: Percent of movies that are rated fresh per month

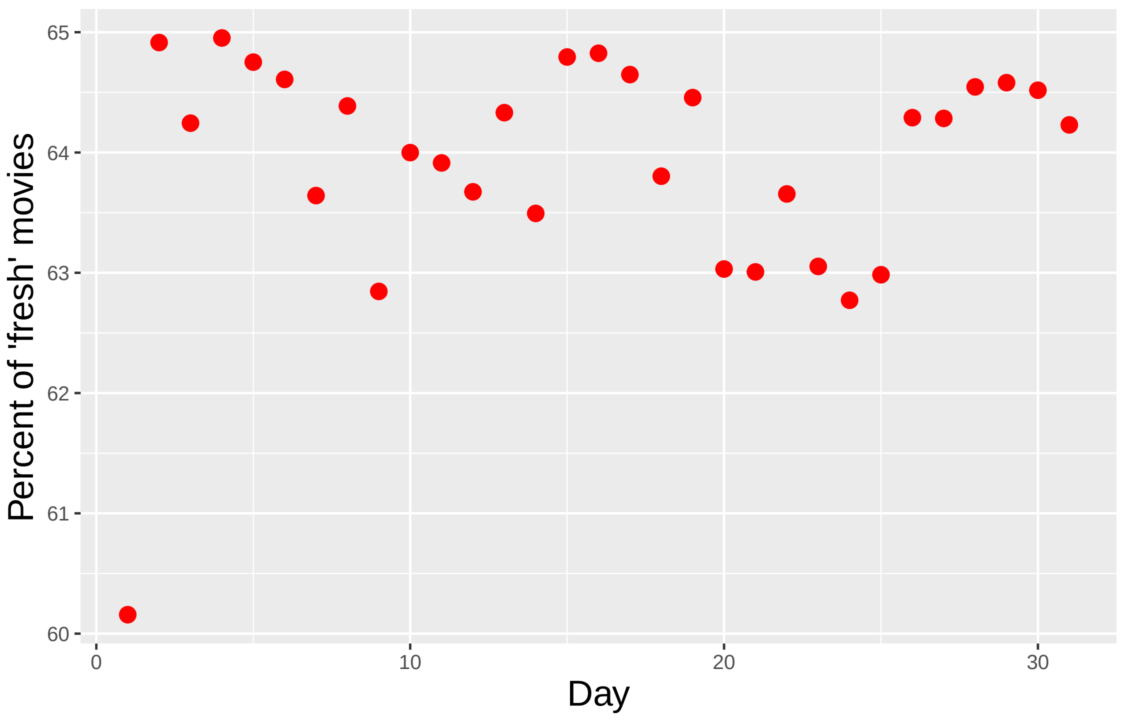


Figure 3: Percent of movies that are rated fresh per day

2. The next aim was to investigate whether differences exist between men and women in the Rotten tomatoes movie dataset, with a particular emphasis on the amount of fresh reviews. The following table shows that overall, men have reviewed more movies (827111) than women (192724) in the Rotten tomatoes movie dataset.

   i

Gender	n
F	192724
M	827111

   ii Here, I plotted the percentage of movies that are rated fresh per year between men and women. The table shows that men gave more fresh movie reviews than women since the year 2000.

review_type	Gender	n
Fresh	F	130065
Fresh	M	518802

   iii Here, I plotted a graph to show the percentage of movie reviews submitted by each gender per year since and including the year 2000 (Fig.4). The red bars show the percentage of reviews submitted by females per year since 2000 (compared with males). Overall, men submitted more movie reviews than women and this difference in the percentage of women submitting reviews since 2000 compared with males has remained fairly constant. In addition I plotted the percentage of reviews rated fresh per year between men and women (Fig.5). Although in the year 2000, males submitted a higher proportion of fresh reviews compared with females, however after the year 2000, females submitted a higher percentage of fresh reviews per year than men. Furthermore, we can observe that this difference in the percentage between males and females seems to be slowly increasing in the last decade.

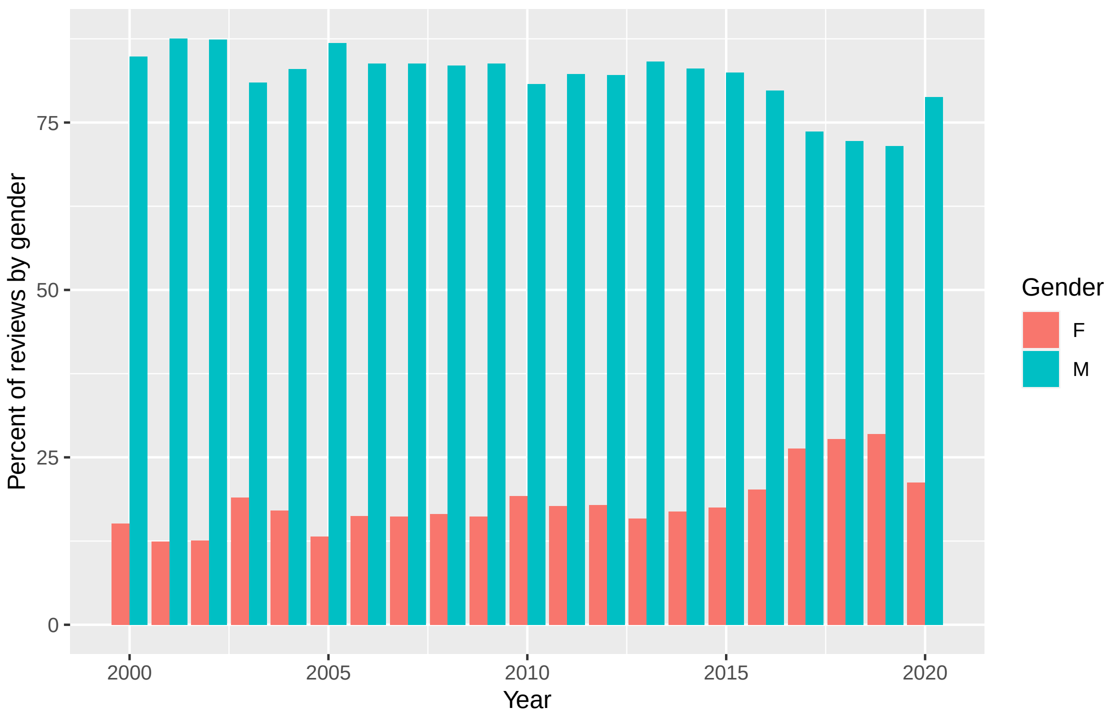


Figure 4: The percent of reviews per year per gender

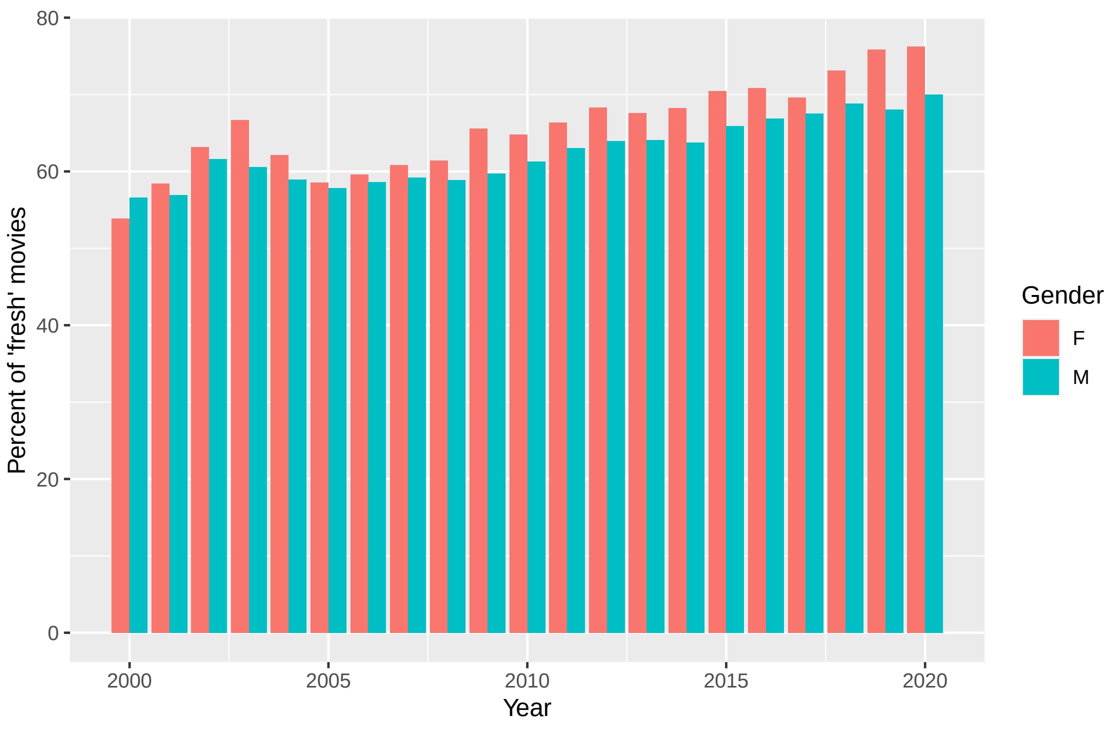


Figure 5: The percent of movies that are rated fresh per year between men and women

3. Here, I sought to assess whether people with names that are more or less common in one country gave more or less favourable reviews.

   i First, I wrote a function called get-countries that takes the file path to the namdict.txt and returns a vector with the countries from the txt file. The vector, that starts with GreatBritain and ends with othercountries has 55 entries (table not shown in PDF).

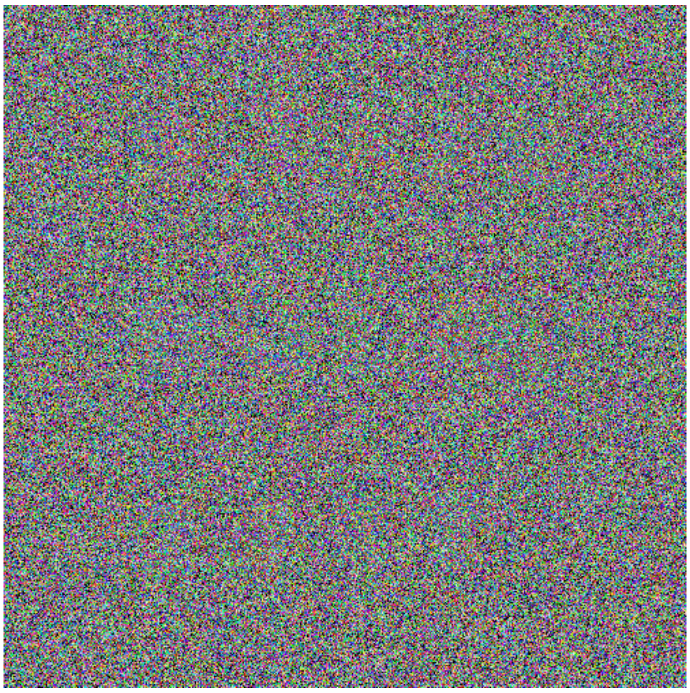
   ii I wrote the get-most-common-country-per-name function, that takes all the available names from namdict.txt and eventually returns a dataframe with the names column and next to it a column with the country where each name is most popular. The name Aadje is most commonly found in East Frisia.

   iii I first created a table using the Rotten tomatoes data and the above function, which contained the country where the first name of each reviewer, for each review, is most popular (table not shown in PDF). I then used the table mentioned, to find what percentage of reviews for each country were fresh or rotten, only showing countries with at least 5000 reviews. The country with the most favourable reviews is Bulgaria with 74.7 percent fresh reviews, whereas Vietnam had 60.7 percent making it the least favourable country (in terms of review types).

Country where most frequent	Review type	n	pct
Vietnam	Fresh	7830	60.7%
EastFrisia	Fresh	56917	61.3%
Luxembourg	Fresh	14504	61.9%
Sweden	Fresh	6529	61.9%
theNetherlands	Fresh	18025	61.9%
Germany	Fresh	6363	62.2%
Portugal	Fresh	16780	62.6%
Italy	Fresh	204898	63.6%
Ireland	Fresh	207903	63.8%
othercountries	Fresh	5198	64%
U.S.A.	Fresh	100768	64.6%
Belgium	Fresh	15081	65.1%
Malta	Fresh	8427	65.5%
Norway	Fresh	24234	66%
Iceland	Fresh	7564	66.6%
France	Fresh	6756	66.7%
India/SriLanka	Fresh	12184	67.3%
China	Fresh	16117	67.7%
Finland	Fresh	8905	69.2%
Azerbaijan	Fresh	7177	69.3%
Bulgaria	Fresh	10835	74.7%

Message from deep space

1. The aim of the second part of this coursework was to denoise an image from a space craft that has taken an image from deep space, which unfortunately due to the nature of transmission of the data to the earth, the resulting signal has been corrupted. First, the PNG image was loaded into R and as shown below it is impossible to decipher the message in the image.



2. Here, a function was written that takes pairs of points, finds those that have the closest RGB values based on L2-norm and calculates their average RGB value.

```
alg1int <- function(A, w1rows, w2cols){

  # For each row and column given,
  # get each point's 3D coordinates [i,j,k] and store them in a list.
  points <- list()
  rows = length(w1rows)
  cols = length(w2cols)
  for (i in 1:rows) {
    for (j in 1:cols) {
      points[[((i-1)*cols+j)]] = A[w1rows[i], w2cols[j], 1:3]
    }
  }

  # Go through all points, using the l2-norm find those points
  # with the closest distance.
  min_dist <- Inf
  index1 <- 0
  index2 <- 0
  n <- rows * cols
  for (i in 1:n) {
    for (j in 1:n) {
      if (i!=j) {
        dist = sqrt(sum( (points[[i]]-points[[j]])^2 ))
        if (dist < min_dist) {
          min_dist <- dist
          index1 <- i
          index2 <- j
        }
      }
    }
  }

  # Calculate the average RGB value
  # of the closest points identified above
  avg = (points[[index1]] + points[[index2]])/2
  return(avg)
}
```

3. A unittest was designed to test whether the alg1int function can indeed calculate the average RGB value for 2 points. Two closest points were selected at random to test this.

```
test_that("Find the average RGB values for 2 closest points", {

  # Set two random closest points
  A1 <- A[3, 6, 1:3]
  A2 <- A[4, 6, 1:3]

  # Calculate the average
  expected_output <- (A1+A2)/2

  func_result <- alg1int(A, 3:4, 6)
  # Compare if results equal
  expect_equal(func_result, expected_output)

})

## Test passed
```

4. Finally, the full algorithm was implemented that takes the noisy 3d RGB image as an input and a smoothing filter of a specific width (w) and uses the alig1int function to calculate the average RGB of the closest pixels and therefore perform the smoothing. It does that for all the rows and columns across the whole 3D image.

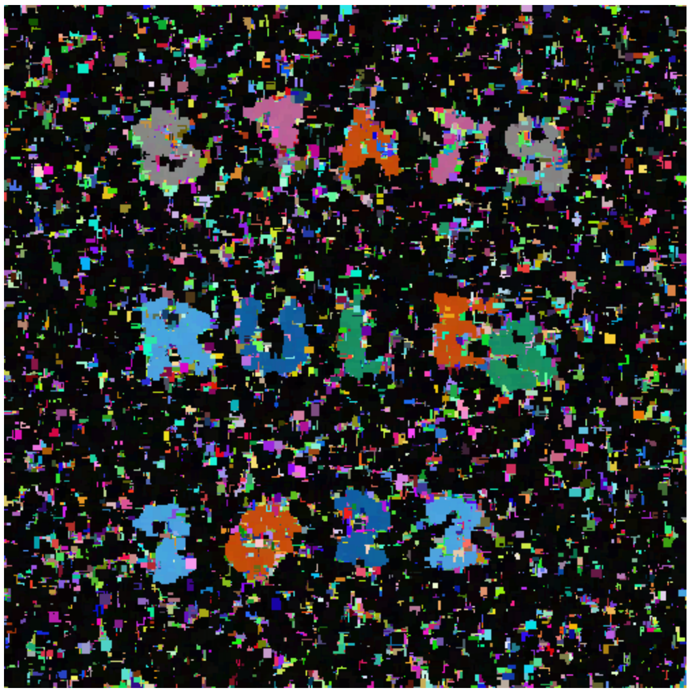
```
algorithm1 <- function(A, w){

  numrows <- nrow(A)
  numcols <- ncol(A)

  # Create empty 3D array for smoothed image
  B <- array(0,dim=c(numrows, numcols, 3))

  # For each iteration, identify the rows of the input image within
  # distance w of i
```

```r
  for(i in 1:numrows){

    w1_start <- i-w+1
    if(w1_start < 1){
      w1_start <- 1
    }

    w1_end <- i+w-1
    if(w1_end > numrows){
      w1_end <- numrows
    }

    # Get the range of rows to use
    w1 <- (w1_start : w1_end)

    # For each iteration, identify the columns of the input image
    # within distance w of j
    for(j in 1:ncol(A)){
      w2_start <- j-w+1
      if(w2_start < 1){
        w2_start <- 1
      }

      w2_end <- j+w-1
      if(w2_end > numrows){
        w2_end <- numrows
      }

      # Get the range of columns to use
      w2 <- (w2_start : w2_end)

      # Use the alg1int function from the previous step
      res <- alg1int(A, w1, w2)

      # Populate the empty array with the new values
      # for the denoised image
      B[i, j, 1:3] <- res
    }
  }
  return(B)
}
```

5. The computational complexity of the ‘alg1int‘ function is $O(n^2)$, where n is the product of the number of rows and columns within the selected window, and is thus dependent on the window size W. This complexity arises from the nested loops used to compute the L2 norm between points in the ‘alg1int‘ function. In the broader ‘algorithm1‘ function, which applies ‘alg1int‘ across the entire 3D array, this complexity is further amplified. Specifically, since ‘algorithm1‘ iterates over each pixel in the $N \times N$ array and applies the ‘alg1int‘ function with a $W \times W$ window at

each iteration, the overall computational complexity escalates to $O(N^2 \cdot W^4)$. This results from the combination of iterating over each element in the $N \times N$ array and the W^4 complexity of the ‘alg1int‘ function at each step, particularly in the worst-case scenario where both the rows and columns influenced by the window size are approximately W. Thus, it’s crucial to recognize that the computational demand of this algorithm scales significantly with both the size of the input array and the chosen window size W.

6. Finally, as requested I apply algorithm1 to mystery.png with w=5 to reveal the hidden message, STATS RULES 2022.


STATS
RULES
2022