Bios 6301: Assignment 2

Jiaheng Yu

Due Tuesday, 19 September, 1:00 PM

50 points total.

Add your name as `author` to the file's metadata section.

Submit a single knitr file (named `homework2.rmd`) by email to marisa.h.blackman@vanderbilt.edu. Place your R code in between the appropriate chunks for each question. Check your output by using the `Knit HTML` button in RStudio.

1. **Working with data** In the `datasets` folder on the course GitHub repo, you will find a file called `cancer.csv`, which is a dataset in comma-separated values (csv) format. This is a large cancer incidence dataset that summarizes the incidence of different cancers for various subgroups. (18 points)

    1. Load the data set into R and make it a data frame called `cancer.df`. (2 points) cancer.df <- read.csv("cancer.csv")
    2. Determine the number of rows and columns in the data frame. (2) dim(cancer.df) # directly reports row x column [1] 42120 8 str(cancer.df) # structure, including names of columns 'data.frame': 42120 obs. of 8 variables: $ year : int 1999 1999 1999 1999 1999 1999 1999 1999 1999 1999 ... $ site : chr "Brain and Other Nervous System" "Brain and Other Nervous System" "Brain and Other Nervous System" "Brain and Other Nervous System" ... $ state : chr "alabama" "alabama" "alabama" "alabama" ... $ sex : chr "Female" "Female" "Female" "Male" ... $ race : chr "Black" "Hispanic" "White" "Black" ... $ mortality : num 0 0 83.7 0 0 ... $ incidence : int 19 0 110 18 0 145 0 0 0 0 ... $ population: int 623475 28101 1640665 539198 37082 1570643 12710 11664 220036 13900 ... 42120 rows, 8 columns
    3. Extract the names of the columns in `cancer.df`. (2) names(cancer.df) [1] "year" "site" "state" "sex"
       [5] "race" "mortality" "incidence" "population"
    4. Report the value of the 3000th row in column 6. (2) cancer.df[3000,6] # data.frame[row, column] [1] 350.69
    5. Report the contents of the 172nd row. (2)
    6. Create a new column that is the incidence *rate* (per 100,000) for each row. The incidence rate is the `(number of cases)/(population at risk)`, which in this case means `(number of cases)/(population at risk) * 100,000`. (3) incidence_rate <- cancer.df*incidence/cancer.df*population cancer.df[,9] <- incidence_rate colnames(cancer.df)[9] <- "incidence rate"
    7. How many subgroups (rows) have a zero incidence rate? (2) colSums(cancer.df==0)[9] # had to input the entire dataframe, then select the 9th column incidence rate 23191
    8. Find the subgroup with the highest incidence rate.(3) > max(cancer.df$"incidence rate") # I can't decide which of these I prefer [1] 0.002611599 > max(cancer.df[,9]) [1] 0.002611599

2. **Data types** (10 points)

1. Create the following vector: `x <- c("5","12","7")`. Which of the following commands will produce an error message? For each command, Either explain why they should be errors, or explain the non-erroneous result. (4 points) > x <- c("5","12","7") > max(x) [1] "7" > sort(x) [1] "12" "5" "7"
2. For the next two commands, either explain their results, or why they should produce errors. (3 points) > y <- c("5",7,12) # warning, then coerced the numeric values to strings > # elements of a vector must all have the same mode, or data type > #y[2] + y[3] # error > y # they were all changed to strings [1] "5" "7" "12"
3. For the next two commands, either explain their results, or why they should produce errors. (3 points)

```
z <- data.frame(z1="5",z2=7,z3=12)
z[1,2] + z[1,3]
```

z <- data.frame(z1="5",z2=7,z3=12) z[1,2] + z[1,3] # row 1, column 2 = 7 [1] 19 # row 1, column 3 = 12 # 7 + 12 = 19 #dtype(z[1,2])

3. **Data structures** Give R expressions that return the following matrices and vectors (*i.e.* do not construct them manually). (3 points each, 12 total)
   1. $(1, 2, 3, 4, 5, 6, 7, 8, 7, 6, 5, 4, 3, 2, 1)$ > x <- seq(1,8,1) > c(x, rev(x[-8])) [1] 1 2 3 4 5 6 7 8 7 6 5 4 3 2 1
   2. $(1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5)$ > # 1 of 1, 2 of 2, etc > # repetitions = 1*1, 2*2, 3*3, 4*4 > rep(1:5, times = 1:5) [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
   3. $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$ > # 0 on the diagonal > matrix1 <- matrix(seq(1, 1, length.out=9), nrow=3, ncol=3) > diag(matrix1) = 0 > matrix1 [,1] [,2] [,3] [1,] 0 1 1 [2,] 1 0 1 [3,] 1 1 0
   4. $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \\ 1 & 16 & 81 & 256 \\ 1 & 32 & 243 & 1024 \end{pmatrix}$ > # first row is numbers > # second row is squares of first row > # third row is cubes of first row, etc > # I know this isn't the most efficient solution, but I wasn't sure how to automatically iterate over the original vector make a matrix (what I originally wanted to do) > > base_vector <- c(1, 2, 3, 4) > matrix2 <- matrix(c(base_vector, base_vector^2, base_vector^3, base_vector^4, base_vector^5), nrow=5, ncol=4, byrow=TRUE) > matrix2 [,1] [,2] [,3] [,4] [1,] 1 2 3 4 [2,] 1 4 9 16 [3,] 1 8 27 64 [4,] 1 16 81 256 [5,] 1 32 243 1024
4. **Basic programming** (10 points)
   1. Let $h(x, n) = 1 + x + x^2 + \ldots + x^n = \sum_{i=0}^{n} x^i$. Write an R program to calculate $h(x, n)$ using a `for` loop. As an example, use `x = 5` and `n = 2`. (5 points) > x <- 5 > n <- 2 > vector <- c(1) > > for (i in 1:n) {

- `new_value <- x^i`
- `vector <- c(vector, new_value)`
- `answer <- sum(vector) # will just give the last value when the loop breaks`
- }

2. If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. Write an R program to perform the following calculations. (5 points)
   a. Find the sum of all the multiples of 3 or 5 below 1,000. (3, euler1) > # 1000/3 = 333.33, call it 333x3 = 999 > # 1000/5 = 200 even, must be less than 1000, call it 199x5 = 995 > # seq() to pick out the multiples, unique() to remove numbers in common (ex: 15 is divisible by 5 and 3), and sum() to add them all up > sum(unique(c(seq(3, 999, 3), seq(5, 995, 5)))) [1] 233168
   b. Find the sum of all the multiples of 4 or 7 below 1,000,000. (2) > # just an extension of the previous problem > # 1000000/4 = 250000 even, 249999x4 = 999996 > # 1000000/7 = 142857.1, call it 142857x7 = 999999 > sum(unique(c(seq(4, 999996, 4), seq(7, 999999, 7)))) [1] 178571071431
3. Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be $(1, 2, 3, 5, 8, 13, 21, 34, 55, 89)$. Write an R program to calculate the sum of the first 15 even-valued terms. (5 bonus points, euler2) > fib <- c(1, 2) # define first two numbers > while (max(length(which(fib%%2 == 0))) < 15) {

- `# keep going until we get 15 even numbers`
- `fib <- c(fib, fib[length(fib) - 1] + fib[length(fib)])`
- } > first_fifteven <- sum(fib[fib%%2 == 0]) > first_fifteven [1] 1485607536

Some problems taken or inspired by projecteuler.