Bios 6301: Assignment 2

Haley Yaremych

Due Tuesday, 21 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 michael.l.williams@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("~/Documents/Fall 2021/Statistical Computing/datasets/cancer.csv", header=TRUE
```

   2. Determine the number of rows and columns in the data frame. (2)

```
nrow(cancer.df)
```

```
## [1] 42120
```

```
ncol(cancer.df)
```

```
## [1] 8
```

   3. Extract the names of the columns in `cancer.df`. (2)

```
colnames(cancer.df)
```

```
## [1] "year"       "site"       "state"      "sex"        "race"
## [6] "mortality"  "incidence"  "population"
```

   4. Report the value of the 3000th row in column 6. (2)

```
cancer.df[3000,6]
```

```
## [1] 350.69
```

   5. Report the contents of the 172nd row. (2)

```
cancer.df[172,]
```

```
##     year                          site  state  sex  race mortality incidence
## 172 1999 Brain and Other Nervous System nevada Male Black         0         0
##     population
## 172      73172
```

   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)

```
cancer.df$rate = (cancer.df$incidence / cancer.df$population)*100000
```

7. How many subgroups (rows) have a zero incidence rate? (2)

```
length(which(cancer.df$rate==0))
```

```
## [1] 23191
```

8. Find the subgroup with the highest incidence rate.(3)

```
which(cancer.df$rate==max(cancer.df$rate))
```

```
## [1] 5797
```

```
cancer.df[5797,]
```

```
##      year     site                 state  sex  race mortality incidence
## 5797 1999 Prostate district of columbia Male Black     88.93       420
##      population     rate
## 5797     160821 261.1599
```

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)

      ```
      max(x)
      sort(x)
      sum(x)
      ```

```
x <- c("5","12","7")
```

max(x) does not produce an error because it can operate on multiple data types. For example, max("a", "b", "c") returns a result. So here, it does not matter that the vector contains characters.

sort(x) can also sort multiple data types, so again it does not matter that we are sorting characters.

sum(x) requires numeric input, so it produces an error because we are attempting to give it characters.

   2. For the next two commands, either explain their results, or why they should produce errors. (3 points)

      ```
      y <- c("5",7,12)
      y[2] + y[3]
      ```

The first line creates a vector with three elements. The second line attempts to add together the second and third elements of y. However, we recieve an error: non-numeric argument to binary operator. This error means that we are attempting to add together two things that R doesn't recognize as numeric. Running `class(y[2])` and `class(y[3])` indicates that R recognizes these elements as characters. Because character is the least flexible data type in R, it looks like the first element being a character has resulted in R coercing the rest of the elements into characters.

   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]
      ```

The first line creates a data frame with three columns containing one element each. Here, we indicate that the z1 column element should be a character, while the z2 and z3 columns contain numeric values. Because data frames allow for each column to be a different data type, this shouldn't be a problem for R and nothing will be coerced into a different type. Running `class(z[1,2])` and `class(z[1,3])` indicates that R is indeed storing these as numeric data types. Thus, we can add these numbers together with no errors.

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. $(1, 2, 3, 4, 5, 6, 7, 8, 7, 6, 5, 4, 3, 2, 1)$

```r
c(seq(1:8), rev(seq(7:1)))
```

```
##  [1] 1 2 3 4 5 6 7 8 7 6 5 4 3 2 1
```

   2. $(1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5)$

```r
rep(c(1:5), times=c(1:5))
```

```
##  [1] 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5
```

   3. $\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$

```r
d = diag(3)
diag(d)=0
d[which(lower.tri(d))] = 1
d[which(upper.tri(d))] = 1
d
```

```
##      [,1] [,2] [,3]
## [1,]    0    1    1
## [2,]    1    0    1
## [3,]    1    1    0
```

   4. $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \\ 1 & 16 & 81 & 256 \\ 1 & 32 & 243 & 1024 \end{pmatrix}$

```r
x = matrix(NA, nrow=5, ncol=4)
x[,1] = 1
x[1,] = seq(1:4)
for(i in 2:5) {x[i,2] = x[i-1,2]*2}
for(i in 2:5) {x[i,3] = x[i-1,3]*3}
for(i in 2:5) {x[i,4] = x[i-1,4]*4}
x
```

```
##      [,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)

```r
h = 0
x = 5
n = 2
```

```r
for(i in 0:n){
  h = h + x^i
}
h
```

```
## [1] 31
```

1. 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)

   1. Find the sum of all the multiples of 3 or 5 below 1,000. (3, euler1)

```r
msum = 0
for (i in 1:1000){
 if (i %% 3 == 0 | i %% 5 == 0){
    msum = msum + i
 }
}
msum
```

```
## [1] 234168
```

```
 2. Find the sum of all the multiples of 4 or 7 below 1,000,000. (2)
```

```r
fsum = 0
for (i in 1:1000000){
 if (i %% 4 == 0 | i %% 7 == 0){
    fsum = fsum + i
 }
}
fsum
```

```
## [1] 178572071431
```

1. 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)

```r
fibs = rep(0,100000)
fibs[1] = 1
fibs[2] = 2

counter = 0
sum = 0
i = 1

while(counter < 15){

  if (i > 2) {
    fibs[i] = fibs[i-2] + fibs[i-1]
  }

  if (fibs[i] %% 2 == 0) {
    sum = sum + fibs[i]
    counter = counter + 1
  }

  i = i + 1
}

sum
```

```
## [1] 1485607536
```

Some problems taken or inspired by projecteuler.