Bios 6301: Assignment 2

Jeongwon Choi

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)

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
setwd("C:/Users/choij33/OneDrive - Vanderbilt/Vanderbilt/Course/Statcomputing/Bios6301-main/Bios6301-ma

getwd()
```

```
## [1] "C:/Users/choij33/OneDrive - Vanderbilt/Vanderbilt/Course/Statcomputing/Bios6301-main/Bios6301-ma
```

```
cancer.df<-read.csv("cancer.csv")
```

2. Determine the number of rows and columns in the data frame. (2)
rows: 42120, cols: 8

```
dim(cancer.df)
```

```
## [1] 42120     8
```

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

```
colnames_cancer.df<-colnames(cancer.df)
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)

```r
print(cancer.df[300, 6])
```

```
## [1] 47.27
```

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

```r
print(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 t

```r
cancer.df2<-cancer.df %>%
  mutate(rate=incidence/population * 100,000)
```

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

```r
  sum(cancer.df2$rate==0)
```

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

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

```r
cancer.df2[cancer.df2$rate == max(cancer.df2$rate), ]
```

```
##      year     site                state  sex  race mortality incidence
## 5797 1999 Prostate district of columbia Male Black     88.93       420
##      population      rate 0
## 5797     160821 0.2611599 0
```

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)
        ```

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

```
## [1] "7"
```

```
sort(x) # "12" "5"  "7"
```

```
## [1] "12" "5"  "7"
```

```
#sum(x) #reason: character type cannot be numerically summed
```

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]
```

```
y <- c("5",7,12)
#y[2] + y[3] #It cannot be calculated because 7 and 12 are considered as character because "5" is a cha
```

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
```

```
##   z1 z2 z3
## 1  5  7 12
```

```
z[1,2] + z[1,3] # It can be calculated because data.frame can hold different types of data - so z2 and
```

```
## [1] 19
```

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

```
c(1:8, 7:1)
```

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

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

```
rep(1:5, 1:5)
```

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

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

```
d<-matrix(data=rep(1,9), nrow=3, ncol=3)
diag(d)<-rep(0, length(diag(d)))
d
```

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

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

```
d4 <- matrix(rep(1:4,4), nrow=4, ncol=4, byrow=TRUE)
for(i in 1:dim(d4)[1]){
  d4[i, ] <- d4[i,]^i
}
d4
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    1    4    9   16
## [3,]    1    8   27   64
## [4,]    1   16   81  256
```

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
a <- 1
for(i in 1:n){
a <- a + x^i
}
a
```

```
## [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 s
```

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

```
a1 <- 0
for (i in 1:999) {
  if (i %% 3 == 0 | i %% 5 == 0) {
    a1 <- a1 + i
  }
}
print(a1)
```

```
## [1] 233168
```

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

```
a2 <- 0
for (i in 1:999999) {
  if (i %% 4 == 0 | i %% 7 == 0) {
    a2 <- a2 + i
  }
}
print(a2)
```

```
## [1] 178571071431
```

1. Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting w

```
fib_seq <- c(1,1)
  for (i in 3:60) {
    fib_seq[i] <- fib_seq[i-1] + fib_seq[i-2]
  }

fib_seq
```

```
##  [1] 1.000000e+00 1.000000e+00 2.000000e+00 3.000000e+00 5.000000e+00
##  [6] 8.000000e+00 1.300000e+01 2.100000e+01 3.400000e+01 5.500000e+01
## [11] 8.900000e+01 1.440000e+02 2.330000e+02 3.770000e+02 6.100000e+02
## [16] 9.870000e+02 1.597000e+03 2.584000e+03 4.181000e+03 6.765000e+03
## [21] 1.094600e+04 1.771100e+04 2.865700e+04 4.636800e+04 7.502500e+04
## [26] 1.213930e+05 1.964180e+05 3.178110e+05 5.142290e+05 8.320400e+05
## [31] 1.346269e+06 2.178309e+06 3.524578e+06 5.702887e+06 9.227465e+06
## [36] 1.493035e+07 2.415782e+07 3.908817e+07 6.324599e+07 1.023342e+08
## [41] 1.655801e+08 2.679143e+08 4.334944e+08 7.014087e+08 1.134903e+09
## [46] 1.836312e+09 2.971215e+09 4.807527e+09 7.778742e+09 1.258627e+10
## [51] 2.036501e+10 3.295128e+10 5.331629e+10 8.626757e+10 1.395839e+11
## [56] 2.258514e+11 3.654353e+11 5.912867e+11 9.567220e+11 1.548009e+12
```

```
counter <-0; limit <- 15; summed <- 0
for (i in 1:length(fib_seq)){
  if (fib_seq[i]%%2==0){
    summed <- summed + fib_seq[i]
    counter <- counter + 1
  }
  if (counter >= limit){
    break
  }
}
summed
```

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

Some problems taken or inspired by projecteuler.