Bios 6301: Assignment 2

Elizabeth Sigworth

due October 10, 2016

(informally) Due Tuesday, 20 September, 1:00 PM

50 points total.

This assignment won't be submitted until we've covered Rmarkdown. Create R chunks for each question and insert your R code appropriately. Check your output by using the `Knit PDF` 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("/var/folders/kp/zlsf12h14y92__lp668ljv2m0000gn/T//RtmpKa1QwJ/data2c844f081265", s
```

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

```
nrow(cancer.df) # number of rows
```

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

```
ncol(cancer.df) # number of columns
```

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

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

```
names(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
## 172 1999 Brain and Other Nervous System nevada Male Black         0
##     incidence population
## 172         0      73172
```

6. Create a new column that is the incidence *rate* (per 100,000) for each row.(3)

```
cancer.df$inc.rate <- cancer.df$incidence/(cancer.df$population/100000)
```

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

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

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

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

```
cancer.df[which(cancer.df$inc.rate == max(cancer.df$inc.rate)),]
```

```
##      year     site                state  sex  race mortality incidence
## 5797 1999 Prostate district of columbia Male Black     88.93       420
##      population inc.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)

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

max(x): This will return "7", since character vectors are sorted based on the first character in the string, so "7" is read as greater than the "1" in the start of "12"

sort(x): This returns the vector in order "12" "5" "7", again sorting on the first character in the string, which would be "1" and then "5" and then "7"

sum(x): This command returns an error, since the sum command cannot be run on type "character".

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

y <- c("5",7,12): This command will create a vector of character type, since R puts all elements of vectors into the same type, so in this case converts 7 and 12 to "7" and "12".

y[2] + y[3]: Because vector y was stored as a character type, attempting to sum these two elements of y will result in an error, since characters cannot be summed.

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): This creates a data frame with one row and three columns; column z1 contains the character "5", column z2 contains the integer 7, and column z3 contains the integer 12.

z[1,2] + z[1,3]: This adds the elements in row 1 column 2 and row 1 column 3, which are the integers 7 and 12, giving a result of 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(seq(1:8),rev(seq(7:1)))
```

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

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

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

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

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

```
mat <- matrix(rep(1,9),nrow=3)
for (i in 1:3) {
  for (j in 1:3) {
    if (i==j) {
      mat[i,j] <- 0
    }
  }
}
mat
```

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

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

```
x <- 1:4
matrix(c(x,x^2,x^3,x^4,x^5),nrow=5,ncol=4,byrow=TRUE)
```

```
##      [,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)

5. 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. (5 points)

```
h.x.n <- function(x,n) {
  h <- 0
  for (i in 0:n) {
    h <- h + x^n
  }
  return(h)
}
```

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)

```
#let a and b be the multiples of interest, and x be the value below which we are interested
sum.mult <- function(a,b,x) {
  total <- 0
  for (i in 1:x-1) {
    if (i%%a==0 | i%%b==0) {
      total <- total + i
    }
  }
  return(total)
}
```

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

```
sum.mult(3,5,1000)
```

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

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

```
sum.mult(4,7,1000000)
```

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

(c) 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)

```
## Generate the first 50 terms of the Fibonacci sequence
fibbo <- NULL
fibbo[1] <- 1
fibbo[2] <- 2
for (i in 3:50) {
  fibbo[i] <- fibbo[i-1] + fibbo[i-2]
}
fibbo
```

```
##  [1]           1           2           3           5           8
##  [6]          13          21          34          55          89
## [11]         144         233         377         610         987
## [16]        1597        2584        4181        6765       10946
## [21]       17711       28657       46368       75025      121393
## [26]      196418      317811      514229      832040     1346269
## [31]     2178309     3524578     5702887     9227465    14930352
## [36]    24157817    39088169    63245986   102334155   165580141
## [41]   267914296   433494437   701408733  1134903170  1836311903
## [46]  2971215073  4807526976  7778742049 12586269025 20365011074
```

```
## Select only even terms from the first 50
fibbo.even <- fibbo[which(fibbo%%2==0)]
fibbo.even
```

```
##  [1]          2          8         34        144        610
##  [6]       2584      10946      46368     196418     832040
## [11]    3524578   14930352   63245986  267914296 1134903170
## [16] 4807526976 20365011074
```

```
length(fibbo.even)
```

```
## [1] 17
```

```
## Since 17 of the first 50 terms were even, we select the first 15 and find the sume of these terms
first.15 <- sum(fibbo.even[1:15])
first.15
```

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

Some problems taken or inspired by projecteuler.