Bios 6301: Assignment 2

Wooyeol Lee

October 7, 2015

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)

2. Load the data set into R and make it a data frame called `cancer.df`. (2 points)

```
setwd("C:/Users/wooyeol/Dropbox/me/coursework/fall2015/statistical computing/homework")
cancer.df <- read.table("cancer.csv", header=T, sep=",")
```

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

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

```
cancer.df1 <- cancer.df[cancer.df$rate==0,]
nrow(cancer.df1)
```

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

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

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

```
##      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)

3. 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)
```

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

```
sort(x)
```

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

```
is.integer("5")
```

```
## [1] FALSE
```

```
is.character("5")
```

```
## [1] TRUE
```

```
"5", "12", "7" are considered as characters, not integers.
max(x):  When alphabetically ordered, their first letters are compared. "7" is the highest.
sort(x): When alphbetically ordered, "1" comes first, "5" and "7" follow that.
sum(x): We cannot sum up the 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]
```

```r
y <- c("5",7,12)
is.character(y[2])
```

```
## [1] TRUE
```

```
y <- c("5",7,12) :All the elements are considered as characters because the elements in a vector
should have the same type and if numerics (7, 12) and a character ("5") are concartenated,
numeric values lose their type.
y[2] + y[3] :We cannot add two 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]
```

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

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

```r
is.numeric(z[1,2])
```

```
## [1] TRUE
```

```
In contrast to vector, a data frame can have elements with different types. Therefore,
the second and third elements are still numeric.
```

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)

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

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

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

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

```r
c(rep(1,1),rep(2,2),rep(3,3),rep(4,4),rep(5,5))
```

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

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

```
matrix(1,nrow=3,ncol=3)-diag(3)
```

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

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

```
outer(c(1:4),c(1:5),'^')
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    1    1    1    1
## [2,]    2    4    8   16   32
## [3,]    3    9   27   81  243
## [4,]    4   16   64  256 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 <- function(x, n) {
  Sum <- 1
  for (i in 1:n) {
    Sum <- Sum + x^i
  }
  return(Sum)
}
```

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)

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

```
x <- c(1:999)
x.3 <- x[x%%3==0]
x.5 <- x[x%%5==0]
x.15 <- x[x%%15==0]
sum(x.3)+sum(x.5)-sum(x.15)
```

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

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

```
x <- c(1:999999)
x.4 <- x[x%%4==0]
x.7 <- x[x%%7==0]
x.28 <- x[x%%28==0]
sum(x.4)+sum(x.7)-sum(x.28)
```

```
## Warning in sum(x.4): integer overflow - use sum(as.numeric(.))

## Warning in sum(x.7): integer overflow - use sum(as.numeric(.))

## Warning in sum(x.28): integer overflow - use sum(as.numeric(.))

## [1] NA
```

```
y <- c(x.4,x.7)
sum(as.numeric(unique(y)))
```

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

%% the answer is not right.%% 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)

```
x<-c(1,2)
even <- 0
count<-0
i <-0
while (TRUE) {
  i <- i+1
  x<-c(x,x[i]+x[i+1])
  if (x[i+1]%%2==0) {
    even<- c(even,x[i+1])
    count<- count+1
  }
  if (count==15) break
}
x
```

```
##  [1]          1          2          3          5          8         13
##  [7]         21         34         55         89        144        233
## [13]        377        610        987       1597       2584       4181
## [19]       6765      10946      17711      28657      46368      75025
## [25]     121393     196418     317811     514229     832040    1346269
## [31]    2178309    3524578    5702887    9227465   14930352   24157817
## [37]   39088169   63245986  102334155  165580141  267914296  433494437
## [43]  701408733 1134903170 1836311903
```

```
even
```

```
##  [1]          0          2          8         34        144        610
##  [7]       2584      10946      46368     196418     832040    3524578
## [13]   14930352   63245986  267914296 1134903170
```

```
sum(even)
```

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

Some problems taken or inspired by projecteuler.