Bios 301: Homework 1

Yu Zhang

September 22, 2014

1. **Working with data** (18 points)

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

```r
cancer.df <- data.frame(read.csv("cancer.csv")) # Read data
```

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

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

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

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

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

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

```r
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)

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

```
## [1] 350.7
```

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

```r
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)

```r
cancer.df["incidence rate (per 100,000)"] <- cancer.df$incidence/cancer.df$population*100000
```

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

```r
sum(cancer.df[,9]==0)
```

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

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

```r
which.max(cancer.df[,9])
```

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

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

   1. `x <- c("5","12","7")` (4 points)

max(x): Since all the variables in the vector are character type, so max() does not campare their value numerically. After many tests, I found out that it actually searches for the biggest digit in a vector. "7" has the biggest digit 7 among all of them ("12" has the biggest digit of 2), so the max() function return "7" as the answer.

sort(x): According to the answer of former question, "7" has the biggest single digit and "12" has the smallest single digit.

sum(x): Error appeared. Since they are all character type variables, they cannot be summed up numerically.

   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]
```

Error appeared. Character is less flexible than numeric, so the type of the vector is character and so are all the variables in the vector. And characters cannot be summed up.

   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]
```

Different from vector, which has only one variable type,a data frame can have a different type for each variable in it. So in this case, z2 and z3 are numeric variables and can be summed up.

3. **Data structures** (3 points each, 12 total)

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

```r
c(seq_len(8),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(1:5,1:5)
```

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

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

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

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

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

```
x <- matrix(1:4, 5, 4, byrow=TRUE)
x[seq(1,5),] <- x[seq(1,5),]**seq(1,5)
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. (5 points)

```
x <- 0.5 #  Could also read from keyboard: x <- as.numeric(readline("enter the value for x:"))
n <- 5 # Could also read from keyboard: as.integer(readline("enter an integer for n:"))
h <- 0
for (i in 0:n){
  h <- h+x**i
}
h
```

```
## [1] 1.969
```

   2. (5 points)

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

```
sum <- 0
for (i in 1:1000-1){
  if ( i%%3 == 0 | i%%5 == 0){
  sum <- sum+i
  }
}
sum
```

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

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

```
i <- 1
j <- 1
x <- 0
sum <- 0
while(TRUE){
 if (4*i <= 7*j){
  x <- 4*i
  i <- i+1
  if (4*i == 7*j){
    j <- j+1
  }
 } else {
  x <- 7*j
  j <- j+1
 }
 if (x>=1000000){break}
 sum <- sum+x
}
sum
```

```
## [1] 1.786e+11
```

3. Write an R program to calculate the sum of the first 15 even-valued terms. (5 bonus points)

```
a <- 1
b <- 2
sum <- b
count <- 1
while(TRUE){
  c <- a+b
  if (c%%2==0){
   sum <- sum+c
   count <- count+1
  }
  if (count==15){break}
  a <- b
  b <- c
}
sum
```

```
## [1] 1.486e+09
```