

Bios 6301: Assignment 2

Rui Wang

September 14, 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)

```
setwd("/Users/ruiwang/Dropbox/Biostatistics/6301/homework/assignment 2")
cancer.df <- read.csv("cancer.csv")
```

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 <- cbind(cancer.df, rate=0)
cancer.df$rate <-cancer.df$incidence/cancer.df$population*100000
```

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

```
nrow(subset(cancer.df, rate == 0))
```

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

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

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

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

```
cancer.df[which.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)

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)
```

`sum(x)` will produce an error message

```
x <- c("5","12","7")
# x consists 3 characters, the values are determined by the first letter or number, 7>5>12
max(x)
```

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

```
# sort functions return the value from min to max by default
sort(x)
```

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

`sum(x)` generates errors,invalid 'type' (character) of argument

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 in $y[2] + y[3]$: non-numeric argument to binary operator `c` function combines character “5” and two numeric argument 7 and 12. The type of each element will be determined by the highest class, which is the character “5”. So 7 and 12 become non-numeric arguments, which can be applied in `+` function.

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[1,2] + z[1,3]
```

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

The result is 19, the summation of the second and third elements. `Data.frame` function is different from `c` function and will keep the class of each element in the frame. So 7 and 12 are numeric arguments, which can be applied in `+` function.

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)

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

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

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

```
# 1
a <- c(1:8,7:1)
a
```

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

```
# 2
b <- rep(1:5, times= 1:5)
b
```

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

```
# 3
c <-matrix(1,nrow=3, ncol=3)
diag(c) <- 0
c
```

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

```
#4
d <- matrix(c(rep(1:4, times=5)), nrow=5, byrow = TRUE)
for (i in 1: nrow(d)) {
  for (j in 1: ncol(d)) {
    d[i,j] <- d[i,j]^i
  }
}
d
```

```
##      [,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 $h(x, n) = 1 + x + x^2 + \dots + x^n = \sum_{i=0}^n x^i$. Write an R program to calculate $h(x, n)$ using a for loop. (5 points)

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

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 all these numbers is 23.

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

```
a <- c(1:999)
sum(a[which(a %% 3 == 0 | a %% 5 == 0)])
```

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

```
i <- 0
s <- 0
while (i < 1000) {
  i <- i + 1
  if (i %% 3 == 0 | i %% 5 == 0) {
    s <- s + i
  }
}
s
```

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

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

```
i <- 0
s <- 0
while (i<1000000) {
  i <- i + 1
  if(i %% 4 == 0 | i %% 7 == 0) {
    s <- s + i
  }
}
s
```

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

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

```
f <- numeric(50)
f[1] <- 1
f[2] <- 2
for (i in 3: 50) {
  f[i] <- f[i-1]+f[i-2]
}
evenf <- f[which(f %% 2 ==0)]
sum(evenf[1:15])
```

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

```
a <- 1
b <- 2
n <- 1
s <- b
while (n<15) {
  c<- a + b
  a <- b
  b <- c
  if (c %% 2==0) {
    s <- s + c
    n <- n+1
  }
}
s
```

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

Some problems taken or inspired by projecteuler.