Computing Assignment 2

Angela Huynh

September 20, 2016

Grade: 53/50

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.

i) Load the data set into R and make it a data frame called cancer.df

```
#cancer.df <- read.csv("~/Documents/Vanderbilt 1/Semester 1/BIOS 6301/Assignments/cancer.csv")
cancer.df <- read.csv("cancer.csv")
#View(cancer.df)
```

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

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

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

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

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

iii) Extract the names of the columns in cancer.df.

```
colnames(cancer.df)
```

```
## [1] "year"       "site"       "state"      "sex"        "race"
## [6] "mortality"  "incidence"  "population"
```

iv) Report the value of the 3000th row in column 6.

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

```
## [1] 350.69
```

v) Report the contents of the 172nd row.

```
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
```

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

```
cancer.df$incidencerate <- cancer.df$incidence/100000
```

JC Grading - 1 For incidence rate above should be incidence / population * 100000

vii) How many subgroups (rows) have a zero incidence rate?

```
length(which(cancer.df$incidencerate==0))
```

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

viii) Find the subgroup with the highest incidence rate.

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

```
## [1] 21387
```

JC Grading - 1 syntax is fine but answer is incorrect b/c of how incidence rate was calculated

2. Data types

i) 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.

```
x <- c("5","12","7")
max(x) #max is listed as "7"
```

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

```
sort(x) #sorts as "12" "5" "7" because the function looks at the first character in the string
```

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

```
#sum(x) #produces an error because these are not numeric types, thus they cannot be summed
```

ii) For the next two commands, either explain their results, or why they should produce errors.

```
y <- c("5",7,12)
y #output is the same as x
```

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

```
#y[2] + y[3] #cannot add strings together
```

iii) For the next two commands, either explain their results, or why they should produce errors.

```
z <- data.frame(z1="5",z2=7,z3=12)
z
```

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

```
z[1,2] + z[1,3] #adds 7 and 12 together, getting 19. Integers are able to be added.
```

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

3. Data structures: Give R expressions that return the following matrices and vectors (i.e. do not construct them manually).

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

```
seq1 <- seq(from=1, to=8, by=1)
seq2 <- seq(from=7, to=1, by=-1)
c(seq1,seq2)
```

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

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

```
c <- vector()
for (i in (1:5)){
  c <- append(c, rep(i,i))
}
c
```

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

iii) $\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$

```
a <- matrix(rep(1,9),3,3)
`diag<-`(a,0)
```

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

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

```
(b <- matrix(c(seq(1,4),seq(1,4)^2,seq(1,4)^3,seq(1,4)^4,seq(1,4)^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

i) 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.

```
hx <- function(x,n){
  valuei <- 0
  sumx <- 0
  for (i in(0:n)){
    sumx <- sumx + x^(i)
    valuei <- valuei + 1
  }
  return(sumx)
}
```

ii) 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.

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

```
multiple35 <- 0
count <- 0
for (i in(1:999)){
  if (i%%3==0 | i%%5==0){
    count <- count + i
    multiple35 <- multiple35 + i
  }
}
multiple35
```

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

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

```
multiple47 <- 0
count1 <- 0
```

```
for (i in(1:999999)){
  if (i%%4==0 | i%%7==0){
    count1 <- count1 + i
    multiple47 <- multiple47 + i
  }
}
multiple47
```

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

iii) 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.

```
countevens <- 1
sumevenfibs <- 2
fibterms <- vector()
evenfib <- vector()
fibterms[1] <- 1
fibterms[2] <- 2
evenfib[1] <- 2

for (i in(3:100)){
  fibterms[i] <- fibterms[i-1] + fibterms[i-2]
  if (fibterms[i]%%2==0){
    evenfib <- append(evenfib, fibterms[i])
    sumevenfibs <- sumevenfibs + fibterms[i]
    countevens <- countevens + 1
    if (countevens >= 15){
      break
    }
  }
}

sumevenfibs
```

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

JC Grading +5 Bonus