Homework1

Yiheng

7/13/2020

1.a.

Make sure current file is in the same folder with folder `data` containing `iowa.csv`. Then use `read.csv` function to load data. Notice that the default value for `sep` in `read.csv()` function is `","`, claim it as `";"`(which the given csv file uses).

```
iowa.df <- read.csv("data/Iowa.csv", header = TRUE, sep = ";")
```

b.

Use `nrow()` and `ncol()` function to read the row number and column number:

```
nrow(iowa.df)
```

```
## [1] 33
```

```
ncol(iowa.df)
```

```
## [1] 10
```

$\Rightarrow$ `iowa.df` has 33 rows and 10 columns.

c.

Use `colnames()` function to read the names of the columns of `iowa.df`:

```
colnames(iowa.df)
```

```
##  [1] "Year"  "Rain0" "Temp1" "Rain1" "Temp2" "Rain2" "Temp3" "Rain3" "Temp4"
## [10] "Yield"
```

d.

Access element in `iowa.df` with `[]`:

```
iowa.df[5,7]
```

```
## [1] 79.7
```

⇒ The value of row 5, column 7 of `iowa.df` is 79.7.

e.

```
iowa.df[2,]
```

```
##   Year Rain0 Temp1 Rain1 Temp2 Rain2 Temp3 Rain3 Temp4 Yield
## 2 1931 14.76  57.5  3.83    75  2.72  77.2   3.3  72.6  32.9
```

2.a.

```
vector1 <- c("5", "12", "7", "32")#1
max(vector1)#2
```

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

```
sort(vector1)#3
```

```
## [1] "12" "32" "5"  "7"
```

<#1> gives an non-erroneous result, because `c()` function combines values into a *vector* or *list*, which can be in type `character`.

<#2> gives the output `"7"`, since the elements of `vector1` are all of type `character`. The help page of `max()` function says:

> Character versions are sorted lexicographically, and this depends on the collating sequence of the locale in use.

<#3>`sort()` function sorts a vector into ascending(default) or descending order. For character type elements, they're also sorted lexicographically.

```
sum(vector1)#4
```

Error in sum(vector1) : invalid 'type' (character) of argument

<#4> should be erroneous, since the arguments of `sum()` function must be numeric or complex or logical vectors, while in this case, it's a character vector.

b.

```r
vector2 <- c("5",7,12)#1
```

```r
vector2[2] + vector2[3] #1
```

Error in vector2[2] + vector2[3] : non-numeric argument to binary operator

<#1>Error comes from applying binary operator + to character type arguments: see the elements of vector2:

```r
vector2
```

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

The output is a vector with elements all of type `character`, since `c()` function coerce all arguments to a common type, which is determined from the highest type of the components in the hierarchy **`NULL`<raw<logical<integer<double<complex<character<list<expression**. In this case, the highest type is `character`.

```r
dataframe3 <- data.frame(z1="5",z2=7,z3=12)#2
dataframe3[1,2] + dataframe3[1,3]#2
```

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

<#2> The result comes from 7+12, since `data.frame` allows columns to have different types, 7 and 12 remain numeric.

```r
list4 <- list(z1="6", z2=42, z3="49", z4=126)
list4[[2]]+list4[[4]]#3
```

```
## [1] 168
```

```r
list4[2]+list4[4]#4
```

Error in list4[2] + list4[4] : non-numeric argument to binary operator

<#3> `list4[[2]]` and `list4[[4]]` are numeric types, as `[[]]` drops names and structures.

<#4> `[]` accesses the elements with names and structures, see output of `list4[2]`:

```r
list4[2]
```

```
## $z2
## [1] 42
```

It's non-numeric, cannot be applied with binary operator +.

3.a.

```
seq(1, 10000, 372)
```

```
##  [1]    1  373  745 1117 1489 1861 2233 2605 2977 3349 3721 4093 4465 4837 5209
## [16] 5581 5953 6325 6697 7069 7441 7813 8185 8557 8929 9301 9673
```

```
seq(1, 10000, length = 50)
```

```
##  [1]     1.0000   205.0612   409.1224   613.1837   817.2449  1021.3061
##  [7]  1225.3673  1429.4286  1633.4898  1837.5510  2041.6122  2245.6735
## [13]  2449.7347  2653.7959  2857.8571  3061.9184  3265.9796  3470.0408
## [19]  3674.1020  3878.1633  4082.2245  4286.2857  4490.3469  4694.4082
## [25]  4898.4694  5102.5306  5306.5918  5510.6531  5714.7143  5918.7755
## [31]  6122.8367  6326.8980  6530.9592  6735.0204  6939.0816  7143.1429
## [37]  7347.2041  7551.2653  7755.3265  7959.3878  8163.4490  8367.5102
## [43]  8571.5714  8775.6327  8979.6939  9183.7551  9387.8163  9591.8776
## [49]  9795.9388 10000.0000
```

b.

`times` as a single integer, the result consists of the whole input repeat this many times;

`each` as a non-negative integer, each element of `x` is repeated `each` times.

See the difference more directly from output:

```
rep(1:3, times=3)
```

```
## [1] 1 2 3 1 2 3 1 2 3
```

```
rep(1:3, each=3)
```

```
## [1] 1 1 1 2 2 2 3 3 3
```

MB.Ch1.2

```
library(DAAG)
```

```
## Loading required package: lattice
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

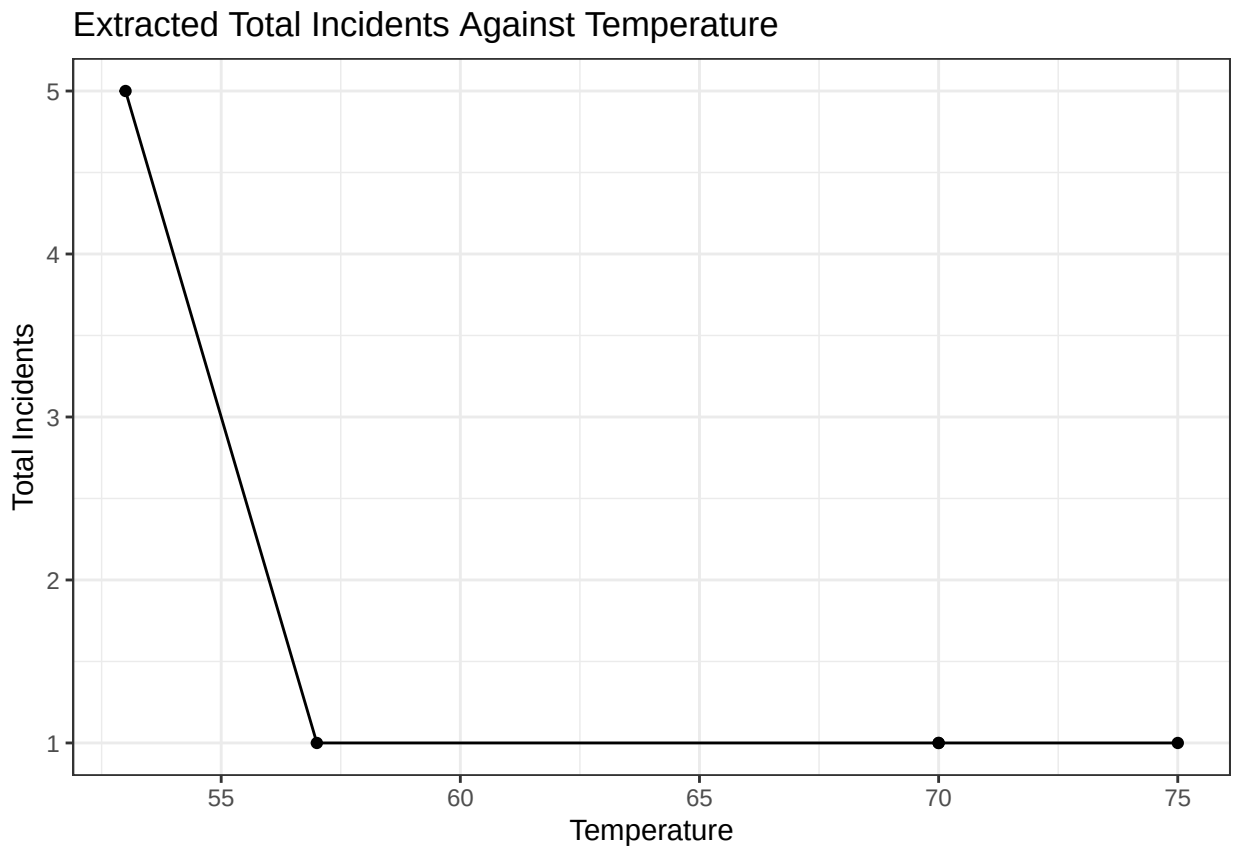
```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Use `rbind()` function to extract rows 1,2,4,11,13,18 from `orings`.

```
extracted_dataframe <- rbind(orings[1,],orings[2,],orings[11,],orings[13,],orings[18,],deparse.level = 
extracted_dataframe
```

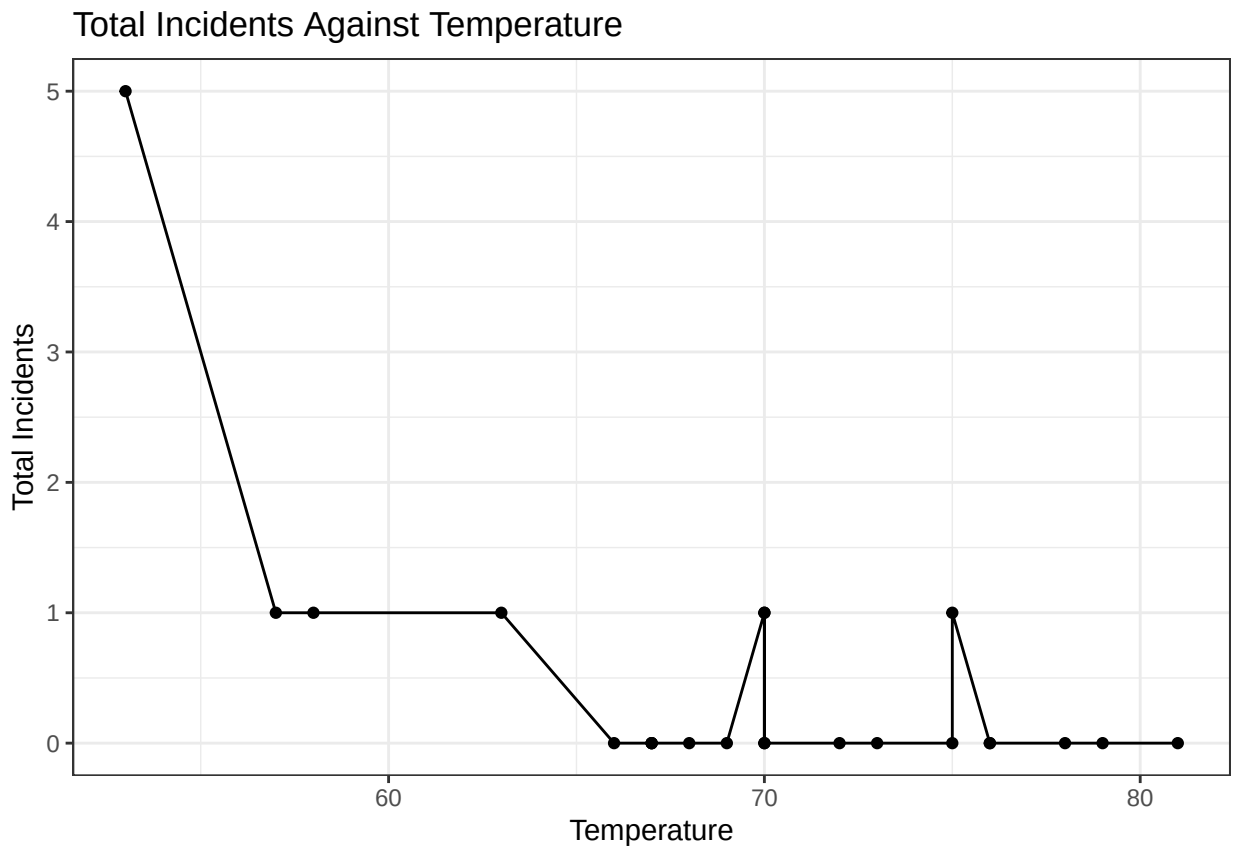
```
##    Temperature Erosion Blowby Total
## 1           53       3      2     5
## 2           57       1      0     1
## 11          70       1      0     1
## 13          70       1      0     1
## 18          75       0      2     1
```

```
ggplot(data = extracted_dataframe) +
  geom_point(aes(x = Temperature,y = Total)) +
  geom_line(aes(x = Temperature, y = Total))+
  labs(y = "Total Incidents",title = "Extracted Total Incidents Against Temperature")+
 theme_bw()
```




```
ggplot(data = orings) +
  geom_point(aes(x = Temperature,y = Total)) +
```

```
 geom_line(aes(x = Temperature, y = Total))+
 labs(y = "Total Incidents",title = "Total Incidents Against Temperature")+
theme_bw()
```



MB.Ch1.4.

a.

```
my_ais <- ais
```

```
str(my_ais)
```

```
## 'data.frame':    202 obs. of  13 variables:
##  $ rcc   : num  3.96 4.41 4.14 4.11 4.45 4.1 4.31 4.42 4.3 4.51 ...
##  $ wcc   : num  7.5 8.3 5 5.3 6.8 4.4 5.3 5.7 8.9 4.4 ...
##  $ hc    : num  37.5 38.2 36.4 37.3 41.5 37.4 39.6 39.9 41.1 41.6 ...
##  $ hg    : num  12.3 12.7 11.6 12.6 14 12.5 12.8 13.2 13.5 12.7 ...
##  $ ferr  : num  60 68 21 69 29 42 73 44 41 44 ...
##  $ bmi   : num  20.6 20.7 21.9 21.9 19 ...
##  $ ssf   : num  109.1 102.8 104.6 126.4 80.3 ...
##  $ pcBfat: num  19.8 21.3 19.9 23.7 17.6 ...
##  $ lbm   : num  63.3 58.5 55.4 57.2 53.2 ...
```

```
##  $ ht    : num  196 190 178 185 185 ...
##  $ wt    : num  78.9 74.4 69.1 74.9 64.6 63.7 75.2 62.3 66.5 62.9 ...
##  $ sex   : Factor w/ 2 levels "f","m": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sport : Factor w/ 10 levels "B_Ball","Field",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Use `is.na()` function to see if there're any missing values:

```
sum(is.na(my_ais))
```

```
## [1] 0
```

⇒ None of the columns hold missing values.

b.

```
my_table<- table(my_ais$sex,my_ais$sport)
```

```
names(which((my_table[1,]/my_table[2,])>2))
```

```
## [1] "Gym"     "Netball"
```

```
names(which((my_table[2,]/my_table[1,])>2))
```

```
## [1] "T_Sprnt" "W_Polo"
```

⇒ In Gym and Netball the factor of female:male is more than 2:1; In T_sprnt and W_Polo the factor of male:female is more than 2:1.

MB.Ch1.6

```
a.Manitoba.lakes <- matrix(c(217,254,248,254,253,227,178,207,217,24387,5374,
                             4624,2247,1353,1223,1151,755,657),ncol = 2)
lakenames <-c("Winnipeg","Winnipegosis","Manitoba","SouthernIndian",
              "Cedar","Island","Gods","Cross","Playgreen")#giving row names
rownames(a.Manitoba.lakes)<- lakenames
colnames(a.Manitoba.lakes) <- c("elevation","area")#giving column names
Manitoba.lakes <- data.frame(a.Manitoba.lakes)#transfer into dataframe structure.
Manitoba.lakes
```

```
##                elevation  area
## Winnipeg             217 24387
## Winnipegosis         254  5374
## Manitoba             248  4624
## SouthernIndian       254  2247
## Cedar                253  1353
## Island               227  1223
## Gods                 178  1151
## Cross                207   755
## Playgreen            217   657
```

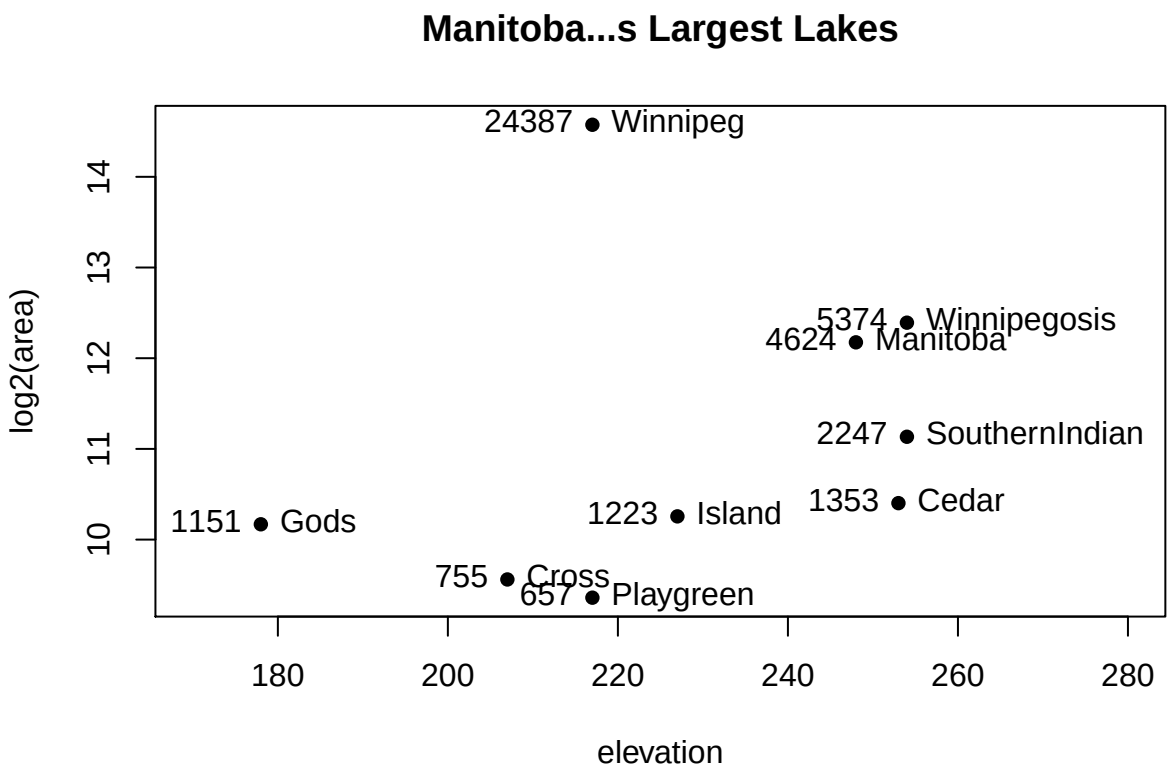
a.

```
attach(Manitoba.lakes)
plot(log2(area) ~ elevation, pch=16, xlim=c(170,280))
# NB: Doubling the area increases log2(area) by 1.0
text(log2(area) ~ elevation, labels=row.names(Manitoba.lakes), pos=4)
text(log2(area) ~ elevation, labels=area, pos=2)
title("Manitoba's Largest Lakes")
```

```
## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <80>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <99>
```



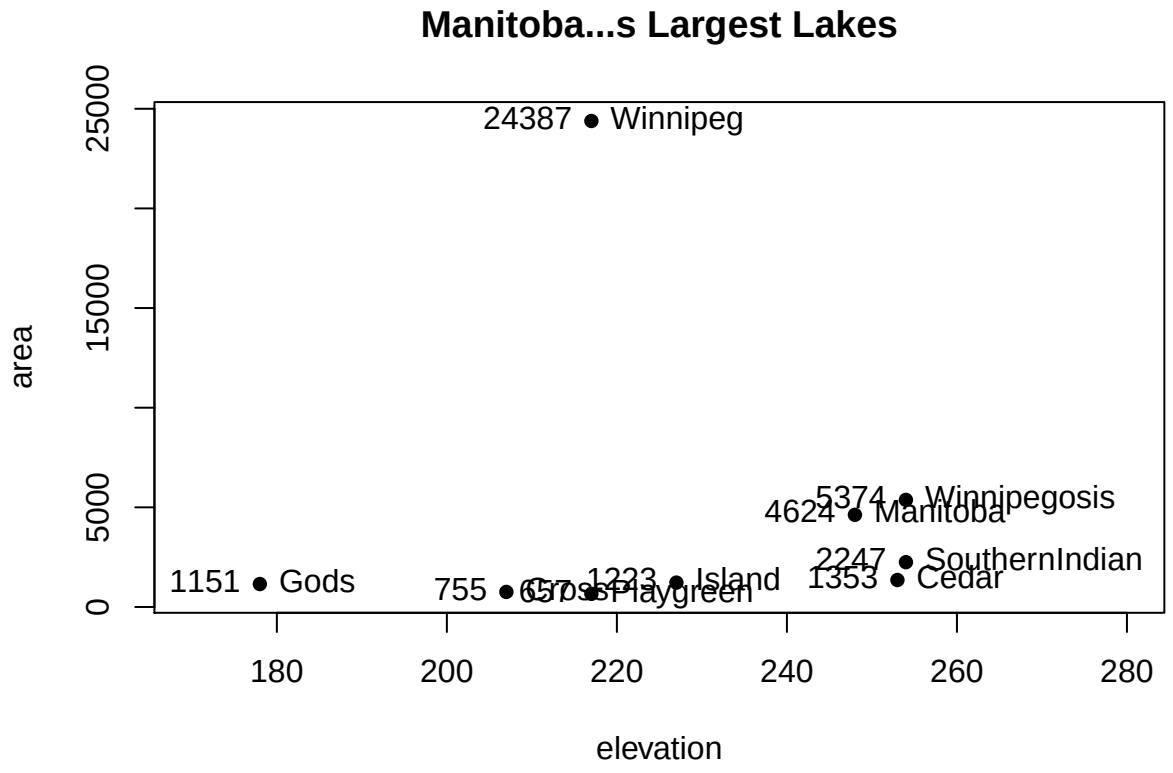
b.

```
plot(area ~ elevation, pch=16, xlim=c(170,280), ylog=T)
text(area ~ elevation, labels=row.names(Manitoba.lakes), pos=4, ylog=T)
text(area ~ elevation, labels=area, pos=2, ylog=T)
title("Manitoba's Largest Lakes")
```

```
## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <80>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on 'Manitoba's
## Largest Lakes' in 'mbcsToSbcs': dot substituted for <99>
```

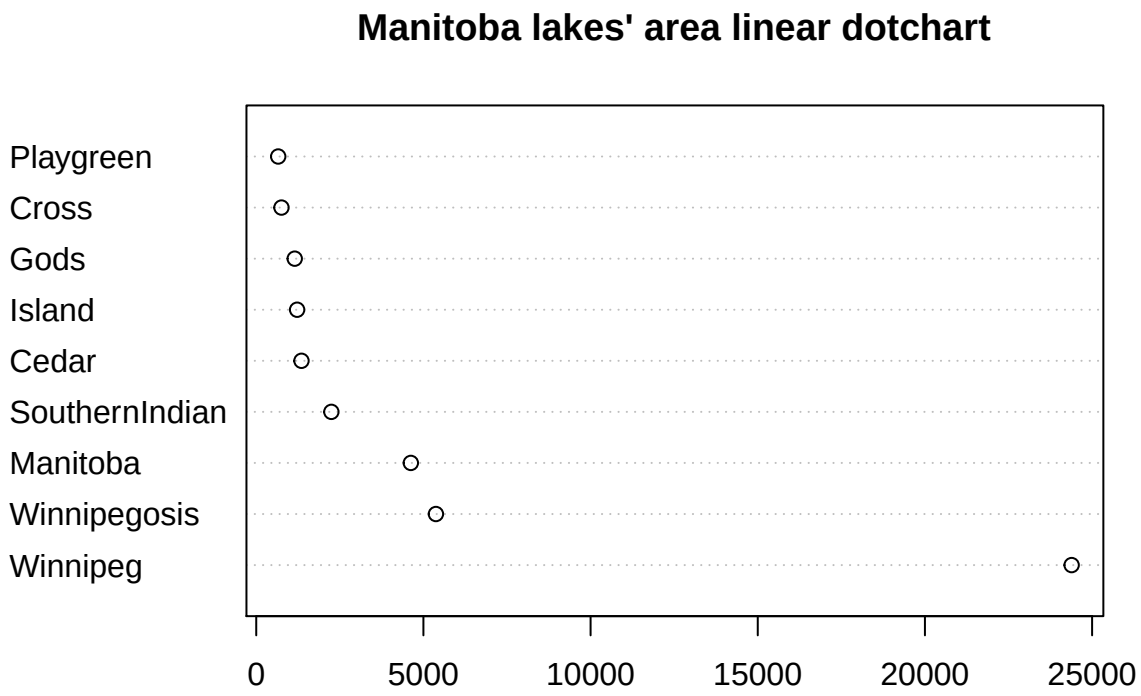


MB.Ch1.7.

a.

Linear scale:

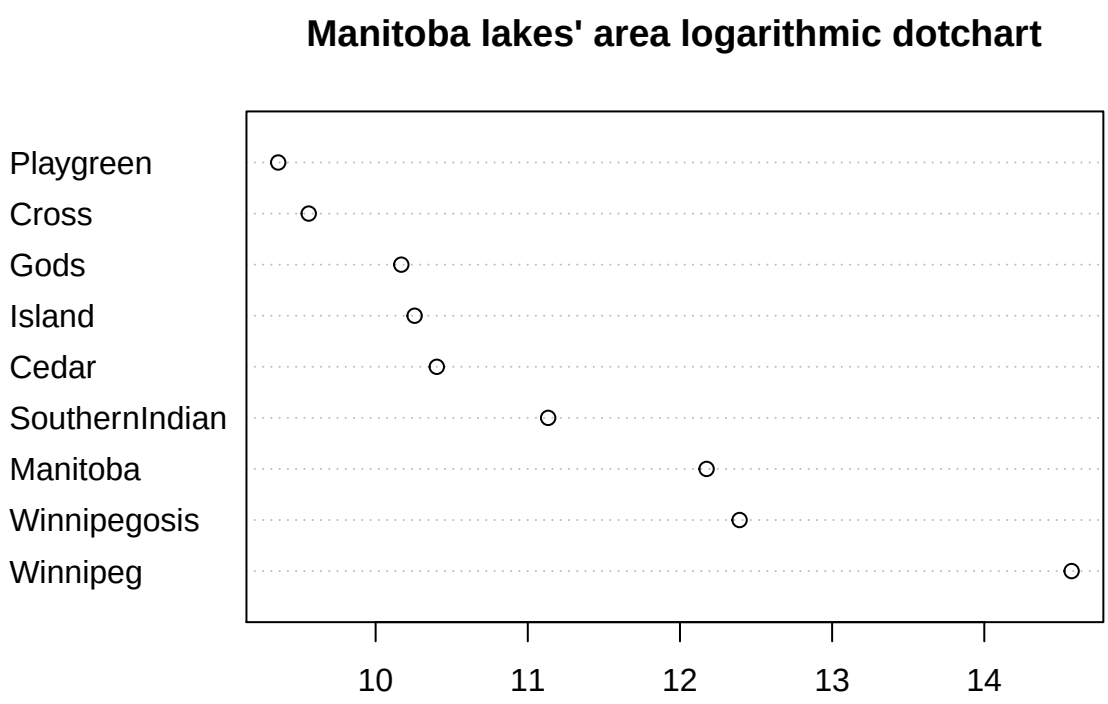
```
dotchart(Manitoba.lakes$area,
         labels = lakenames,
         main = "Manitoba lakes' area linear dotchart")
```



b.

Logarithmic scale:

```
dotchart(log2(Manitoba.lakes$area),
         labels = lakenames,
         main = "Manitoba lakes' area logarithmic dotchart")
```



MB.Ch1.8.

```
sum(Manitoba.lakes$area)
```

```
## [1] 41771
```

⇒ A lower bound for the area of Manitoba covered by water is 41771 square kilometers.