《数学实践》作业一

许乐乐

1. The Iowa data set iowa.csv is a toy example that summarises the yield of wheat (bushels per acre) for the state of Iowa between 1930-1962. In addition to yield, year, rainfall and temperature were recorded as the main predictors of yield.
    a. First, we need to load the data set into R using the command `read.csv()`. Use the help function to learn what arguments this function takes. Once you have the necessary input, load the data set into R and make it a data frame called `iowa.df`.
    b. How many rows and columns does `iowa.df` have?
    c. What are the names of the columns of `iowa.df`?
    d. What is the value of row 5, column 7 of `iowa.df`?
    e. Display the second row of `iowa.df` in its entirety.

```
# ?read.csv ## Use the help function to learn what arguments this function takes.
library(tidyverse)
library(DAAG)
iowa.df<-read.csv("data/Iowa.csv",header=T,sep=";")
dim(iowa.df)
```

```
## [1] 33 10
```

Solution: iowa.df has 33 rows and 10 columns.

```
names(iowa.df)
```

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

Solution: the names of the columes of iowa.df are [1] "Year" "Rain0" "Temp1" "Rain1" "Temp2" [6] "Rain2" "Temp3" "Rain3" "Temp4" "Yield"

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

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

```
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. Syntax and class-typing.
   a. For each of the following commands, either explain why they should be errors, or explain the non-erroneous result.

```
vector1 <- c("5", "12", "7", "32")
max(vector1)
sort(vector1)
sum(vector1)
```

```
b. For the next series of commands, either explain their results, or why they should produce erro
```

```
vector2 <- c("5",7,12)
vector2[2] + vector2[3]

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

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

```r
vector1 <- c("5", "12", "7", "32")
max(vector1)
```

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

```r
sort(vector1)
```

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

```r
# sum(vector1)
# Error in sum(vector1) : 'type'(character) 参数不对
```

vector1 是一个字符串向量。因为向量中的元素是字符串，所以在比较大小时，会先按照第一个字符的大小进行排序，所以在求最大值 max() 和排序 sort() 时可以正常执行。而字符串无法按照数字格式进行求和，因此求和 sum() 不能正常执行。

```r
vector2 <- c("5",7,12)
# vector2[2] + vector2[3]
# Error in vector2[2] + vector2[3] : 二进列运算符中有非数值参数
```

```r
vector2[2]
```

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

```r
is.character(vector2[2])
```

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

vector2 的第一个元素是字符串格式，在同一个向量 vector 中各个元素的类型需要保持一致，因此第二、三个元素也为字符串格式。所以不能像数字格式那样进行加减运算。

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

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

```
dataframe3[1,1]
```

```
## [1] "5"
```

```
is.character(dataframe3[1,1])
```

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

```
dataframe3[1,2]
```

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

```
is.character(dataframe3[1,2])
```

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

数据框 dataframe 允许内部元素的类型不同。因此 dataframe3 中第一个元素是字符串；第二个元素和第三个元素是数值，可以进行加减运算。

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

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

```
# list4[2]+list4[4]
# Error in list4[2] + list4[4] : 二进列运算符中有非数值参数
```

```
list4[1]
```

```
## $z1
## [1] "6"
```

```
list4[[1]]
```

```
## [1] "6"
```

列表 list 允许内部元素的类型不一样。用双括号 [[]] 取元素时，得到的是元素本身；用单括号 [] 取元素时，得到的是元素名和元素本身。因此元素本身为数值型的可以进行加减运算，而元素名和元素本身的结合体无法进行加减运算。

3. Working with functions and operators.
   a. The colon operator will create a sequence of integers in order. It is a special case of the function `seq()` which you saw earlier in this assignment. Using the help command `?seq` to learn about

the function, design an expression that will give you the sequence of numbers from 1 to 10000 in increments of 372. Design another that will give you a sequence between 1 and 10000 that is exactly 50 numbers in length.

b. The function `rep()` repeats a vector some number of times. Explain the difference between 'rep(1:3, times=3) and rep(1:3, each=3).

```
seq(1,10000,by=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.out=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
```

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

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

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

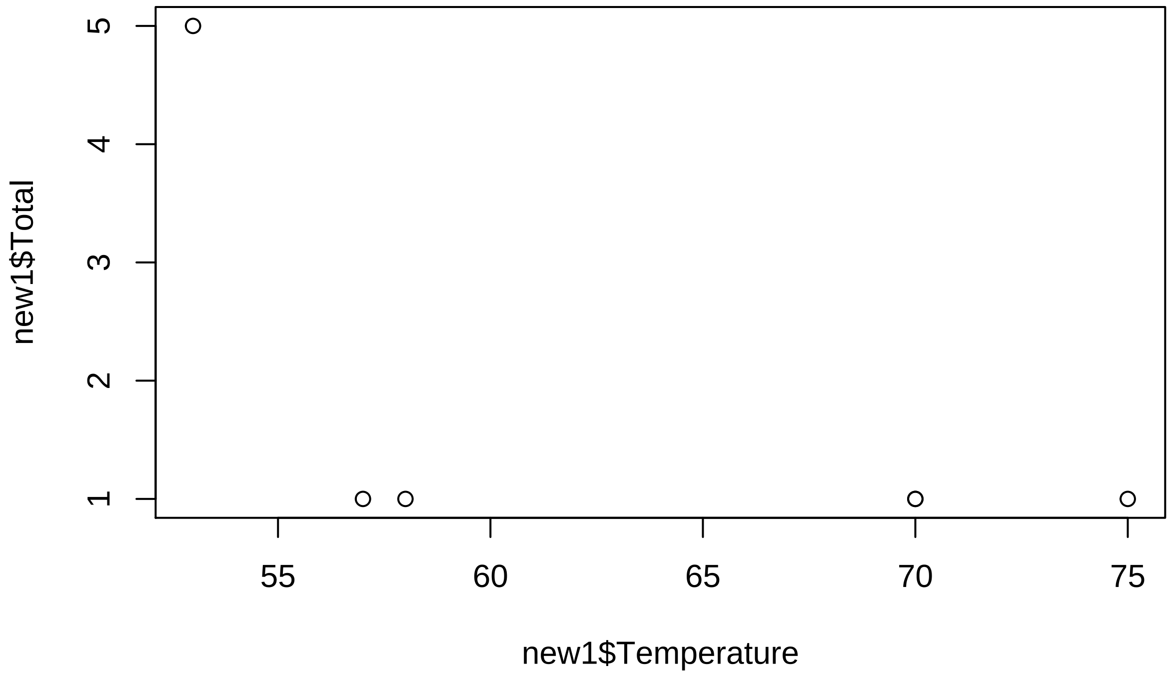
```
## [1] 1 1 1 2 2 2 3 3 3
```

times=3 是将所有元素整体循环三次；each=3 是将逐个元素循环三次。

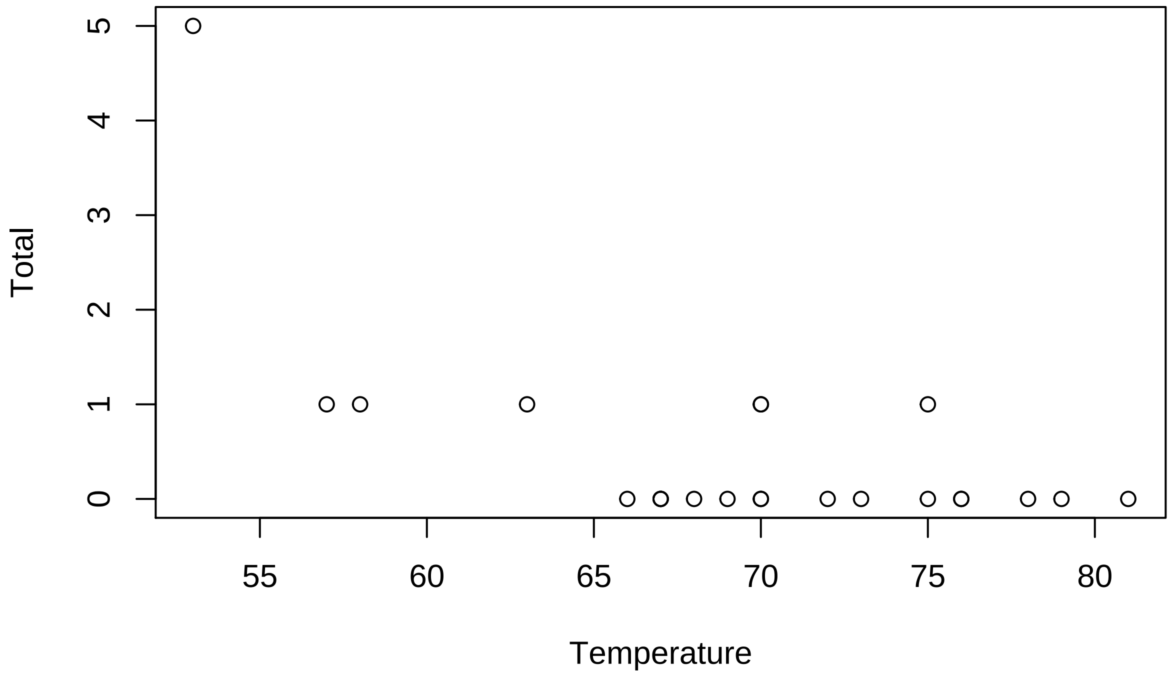
MB.Ch1.2. The orings data frame gives data on the damage that had occurred in US space shuttle launches prior to the disastrous Challenger launch of 28 January 1986. The observations in rows 1, 2, 4, 11, 13, and 18 were included in the pre-launch charts used in deciding whether to proceed with the launch, while remaining rows were omitted.

Create a new data frame by extracting these rows from orings, and plot total incidents against temperature for this new data frame. Obtain a similar plot for the full data set.

```
attach(orings)
new1<-orings[c(1,2,3,11,13,18),]
plot(new1$Temperature,new1$Total)
```




```
plot(Temperature,Total)
```



MB.Ch1.4. For the data frame ais (DAAG package)

(a) Use the function `str()` to get information on each of the columns. Determine whether any of the columns hold missing values.

```
str(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 ...
```

```
which(colSums(is.na(ais))>0)
```

```
## named integer(0)
```

ais 没有缺失值。

(b) Make a table that shows the numbers of males and females for each different sport. In which sports is there a large imbalance (e.g., by a factor of more than 2:1) in the numbers of the two sexes?

```
attach(ais)
table(sport,sex)
```

```
##          sex
## sport      f  m
##   B_Ball  13 12
##   Field    7 12
##   Gym      4  0
##   Netball 23  0
##   Row     22 15
##   Swim     9 13
##   T_400m  11 18
##   T_Sprnt  4 11
##   Tennis   7  4
##   W_Polo   0 17
```

```
rate<-table(sport,sex)[,1]/table(sport,sex)[,2]
rate
```

```
##    B_Ball     Field       Gym   Netball       Row      Swim    T_400m   T_Sprnt
## 1.0833333 0.5833333       Inf       Inf 1.4666667 0.6923077 0.6111111 0.3636364
##    Tennis    W_Polo
## 1.7500000 0.0000000
```

```
subset(rate, rate<0.5| rate>2)
```

```
##       Gym   Netball   T_Sprnt    W_Polo
##       Inf       Inf 0.3636364 0.0000000
```

有重大性别差异的运动：Gym Netball T_Sprnt W_Polo

MB.Ch1.6.Create a data frame called Manitoba.lakes that contains the lake's elevation (in meters above sea level) and area (in square kilometers) as listed below. Assign the names of the lakes using the `row.names()` function.

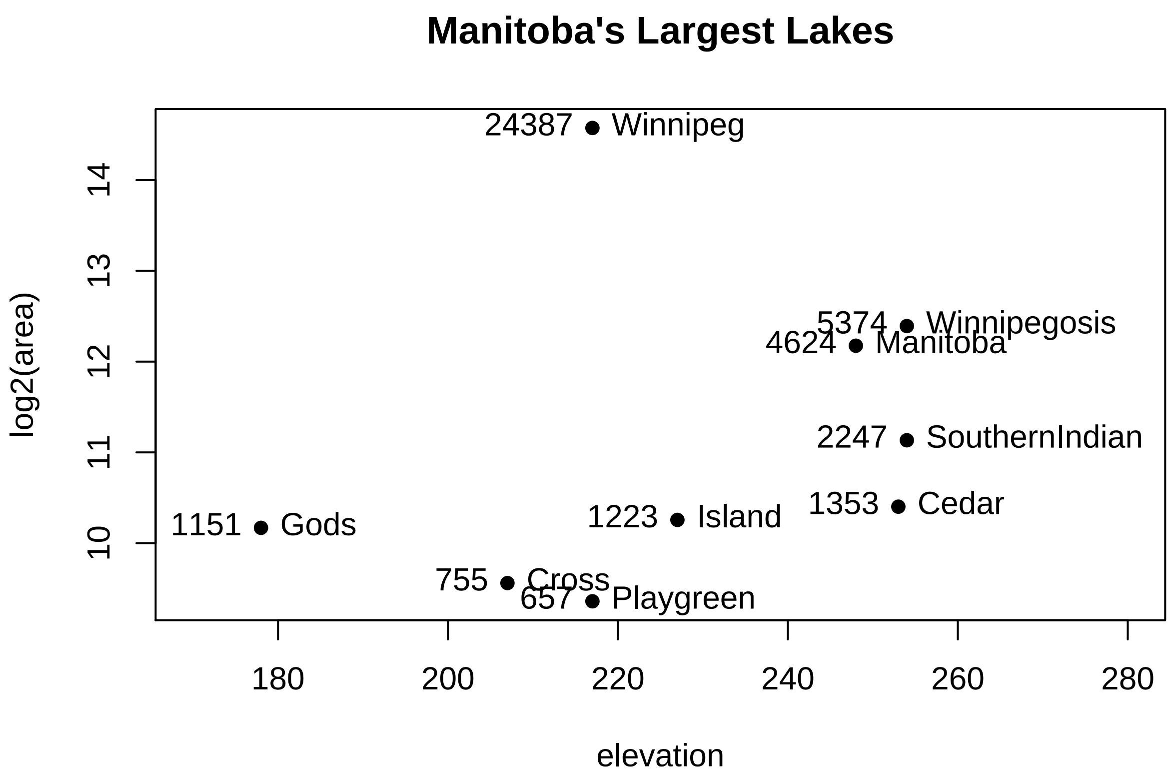
```
row.names(Manitoba.lakes)
```

```
## [1] "Winnipeg"       "Winnipegosis"   "Manitoba"       "SouthernIndian"
```

```
## [5] "Cedar"          "Island"         "Gods"           "Cross"
## [9] "Playgreen"
```

(a) Use the following code to plot `log2(area)` versus elevation, adding labeling information (there is an extreme value of area that makes a logarithmic scale pretty much essential):

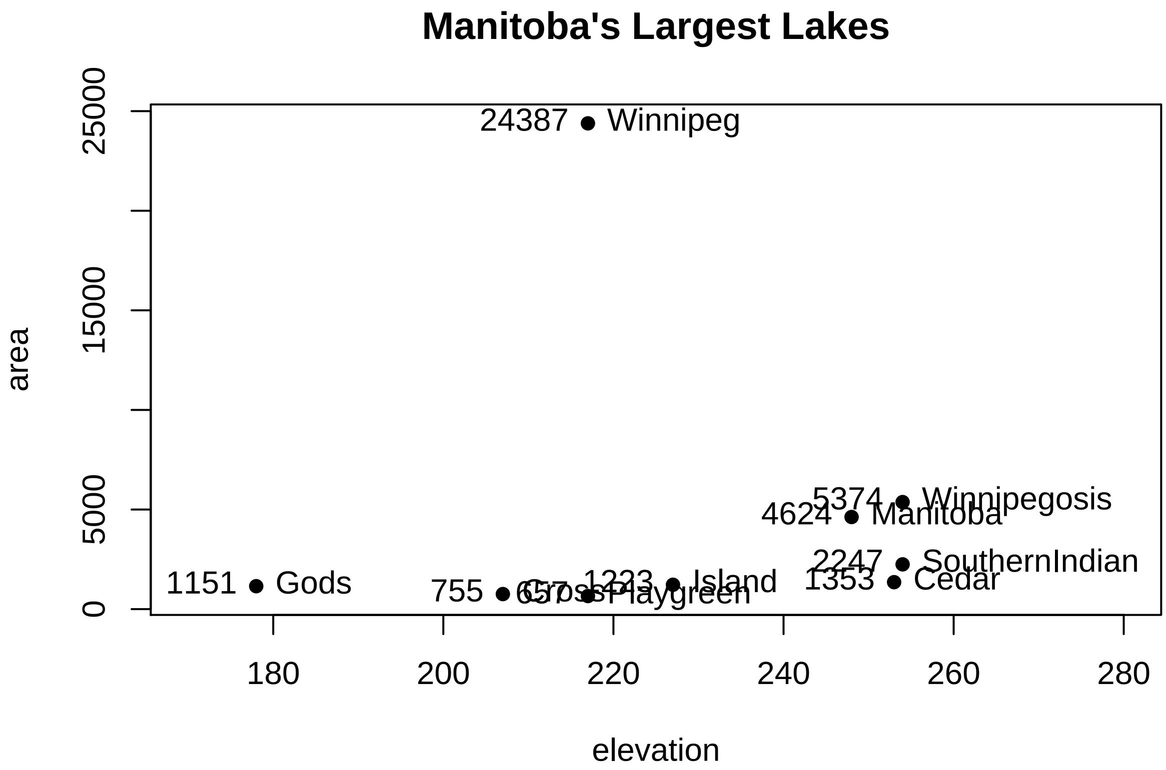
```
attach(Manitoba.lakes)
plot(log2(area) ~ elevation, pch=16, xlim=c(170,280))
# NB: Doubling the area increases log2(area) by 1.0
text(log2(area) ~ elevation, labels=row.names(Manitoba.lakes), pos=4)
text(log2(area) ~ elevation, labels=area, pos=2)
title("Manitoba's Largest Lakes")
```



Devise captions that explain the labeling on the points and on the y-axis. It will be necessary to explain how distances on the scale relate to changes in area.

(b) Repeat the plot and associated labeling, now plotting area versus elevation, but specifying `ylog=TRUE` in order to obtain a logarithmic y-scale.

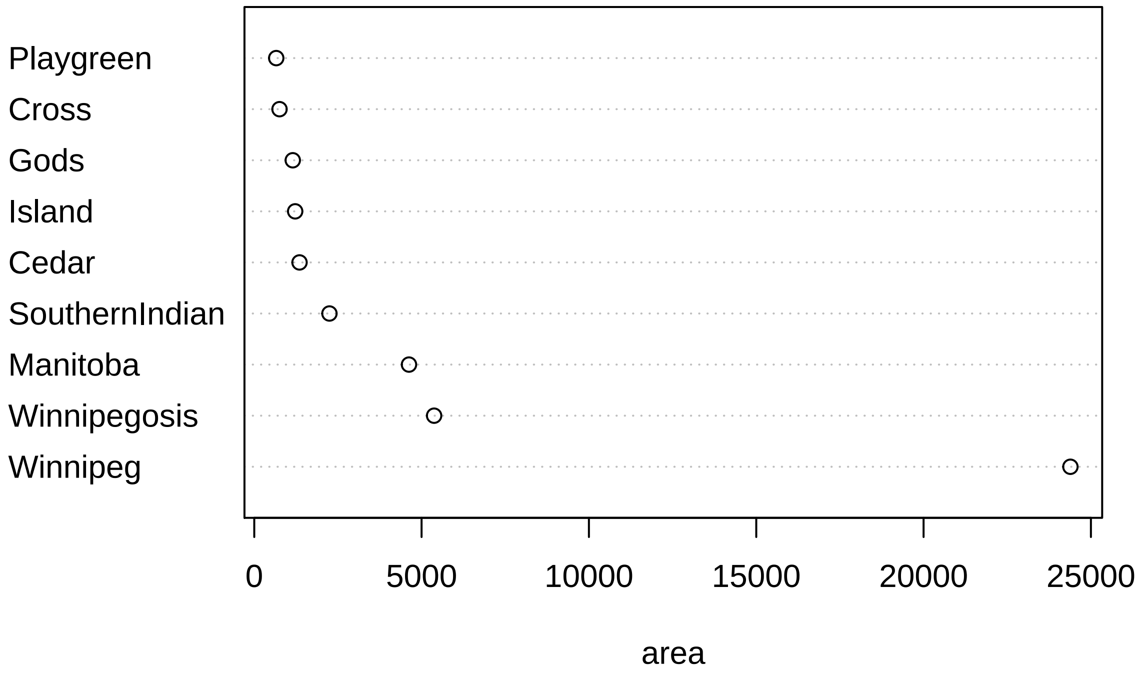
```
plot(area ~ elevation, pch=16, xlim=c(170,280), ylog=T)
text(area ~ elevation, labels=row.names(Manitoba.lakes), pos=4, ylog=T)
text(area ~ elevation, labels=area, pos=2, ylog=T)
title("Manitoba's Largest Lakes")
```



MB.Ch1.7. Look up the help page for the R function `dotchart()`. Use this function to display the areas of the Manitoba lakes (a) on a linear scale, and (b) on a logarithmic scale. Add, in each case, suitable labeling information.

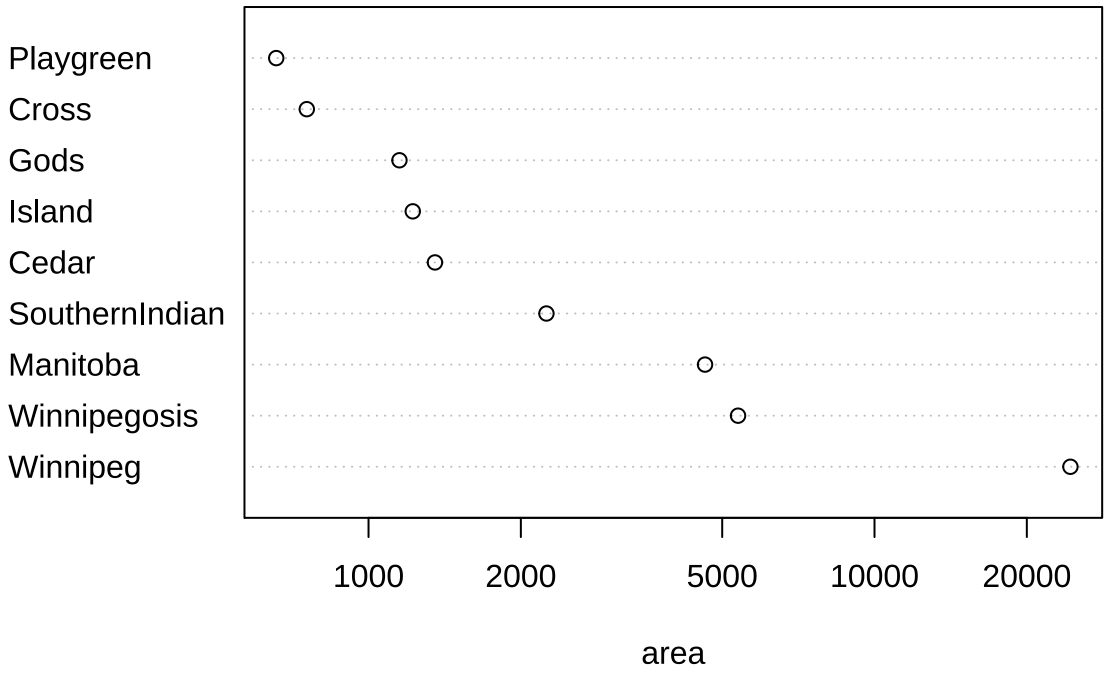
(a) on a linear scale

```
dotchart(Manitoba.lakes$area,labels=row.names(Manitoba.lakes),xlab="area")
```



(b) on a logarithmic scale

```
dotchart(Manitoba.lakes$area,labels=row.names(Manitoba.lakes),xlab="area",log="x")
```



MB.Ch1.8. Using the `sum()` function, obtain a lower bound for the area of Manitoba covered by water.

```
sum(Manitoba.lakes[,"area"])
```

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