Homework 1

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. ## T1.a

```
iowa.df <- read.csv("data/Iowa.csv", sep = ';', header=T)
dim(iowa.df)
```

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

T1.b

```
dim(iowa.df)
```

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

T1.c

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

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

T1.d

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

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

T1.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. 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 errors.
```

```
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]
```

T2

- (a)
- Vector1 is is a character vector.max(vector1) and sort(vector1) will compare in lexicographical order,but sum(vector1) will Will produce an error.
- Because characters can be sorted in lexicographical order, but cannot be summed directly.
- The results are `"7"`,`c("12","32","5","7")`,`Error in sum(vector1) : invalid 'type' (character) of argument` in turn.
- (b)
- `vector2[2] + vector2[3]`:It will generate an error. Single brackets `[...]` are used to subset a list, and the result is still a list. This command tries to add two lists (`list(z2=42)` and `list(z4=126)`), which is not allowed. The error message will be `Error in list4[2] + list4[4] : non-numeric argument to binary operator`.
- `dataframe3[1,2] + dataframe3[1,3]`:The result is `19`. In a data frame, each column can have a different data type. The value of `dataframe3[1,2]` is the numeric `7`, and the value of `dataframe3[1,3]` is the numeric `12`. They are both of type `numeric` and can be added normally.
- `list4[[2]]+list4[[4]]`:The result is `168`. Double brackets `[[...]]` are used to extract a single element from a list. Here, the second element (value `42`) and the fourth element (value `126`) are extracted and can be added normally.
- `list4[2]+list4[4]`:It will produce an error. Single brackets `[...]` are used to subset a list, and the result is still a list. This command tries to add two lists (`list(z2=42)` and `list(z4=126)`), which is not allowed. The error message will be `Error in list4[2] + list4[4] : non-numeric argument to binary operator`.

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)`.

T3.a

```
seq(from = 1,to = 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
```

```r
seq(from = 1,to = 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
```

T3.b

- `rep(1:3, times = 3)`: This command will repeat the entire vector `c(1, 2, 3)` three times. The result is `c(1, 2, 3, 1, 2, 3, 1, 2, 3)`.
- `rep(1:3, each = 3)`: This command will repeat each element in the vector three times, and then proceed to the next element. The result is `c(1, 1, 1, 2, 2, 2, 3, 3, 3)`.

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.

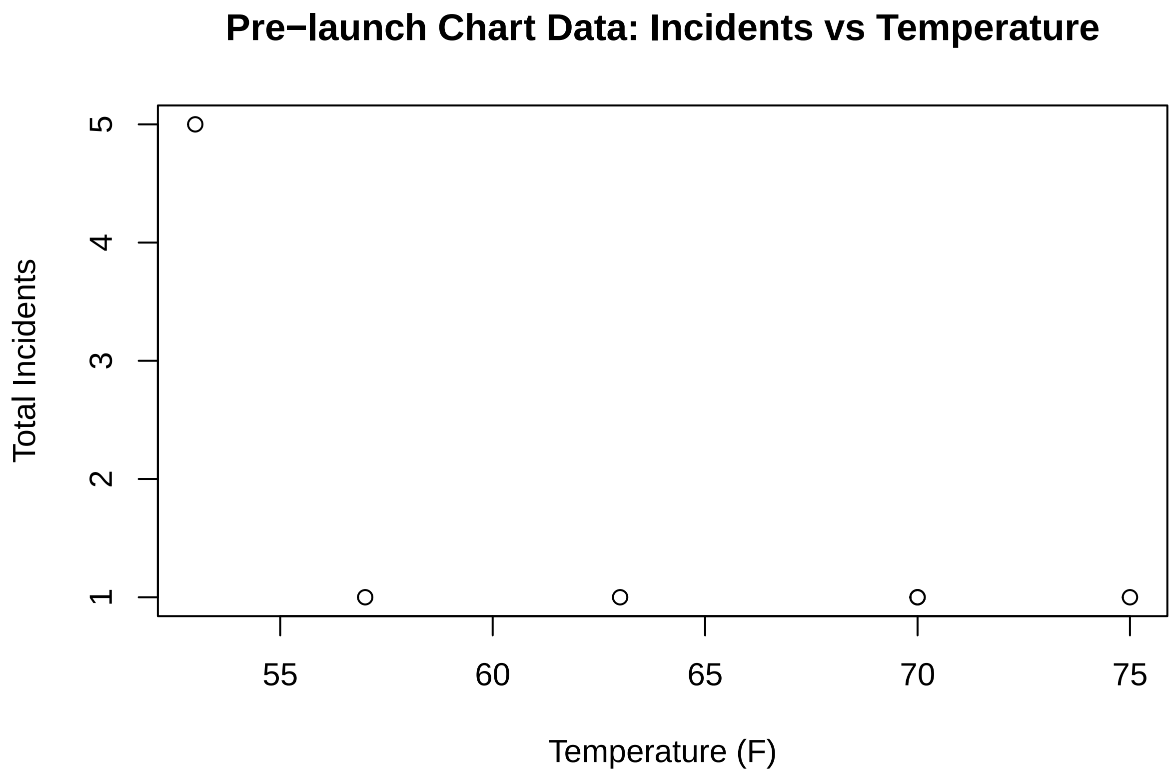
Create a new data frame by extracting these rows from orings, and plot total incidents against temperature for this new data frame. Obtain a similar plot for the full data set.

Ch1.2

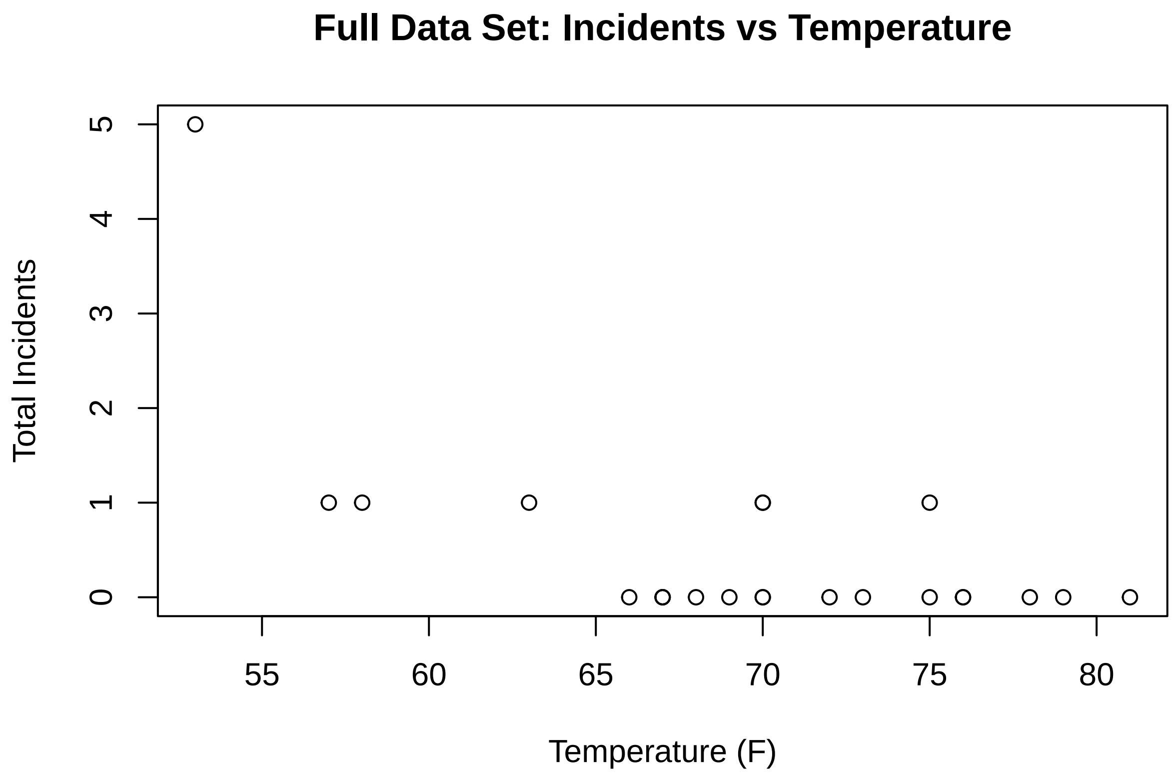
```r
data(orings)

challenger_rows <- c(1,2,4,11,13,18)
orings_subset <- orings[challenger_rows,]

plot(Total ~ Temperature,data = orings_subset,
     main = "Pre-launch Chart Data: Incidents vs Temperature",
     xlab = "Temperature (F)", ylab = "Total Incidents")
```




```
plot( Total ~ Temperature, data = orings,
      main = "Full Data Set: Incidents vs Temperature",
      xlab = "Temperature (F)", ylab = "Total Incidents")
```



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.

(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?

Ch1.4

```
data(ais)
# (a)
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 ...
```

```r
summary(ais)
```

```
##       rcc             wcc              hc              hg
##  Min.   :3.800   Min.   : 3.300   Min.   :35.90   Min.   :11.60
##  1st Qu.:4.372   1st Qu.: 5.900   1st Qu.:40.60   1st Qu.:13.50
##  Median :4.755   Median : 6.850   Median :43.50   Median :14.70
##  Mean   :4.719   Mean   : 7.109   Mean   :43.09   Mean   :14.57
##  3rd Qu.:5.030   3rd Qu.: 8.275   3rd Qu.:45.58   3rd Qu.:15.57
##  Max.   :6.720   Max.   :14.300   Max.   :59.70   Max.   :19.20
##
##       ferr             bmi             ssf             pcBfat
##  Min.   :  8.00   Min.   :16.75   Min.   : 28.00   Min.   : 5.630
##  1st Qu.: 41.25   1st Qu.:21.08   1st Qu.: 43.85   1st Qu.: 8.545
##  Median : 65.50   Median :22.72   Median : 58.60   Median :11.650
##  Mean   : 76.88   Mean   :22.96   Mean   : 69.02   Mean   :13.507
##  3rd Qu.: 97.00   3rd Qu.:24.46   3rd Qu.: 90.35   3rd Qu.:18.080
##  Max.   :234.00   Max.   :34.42   Max.   :200.80   Max.   :35.520
##
##       lbm              ht              wt            sex         sport
##  Min.   : 34.36   Min.   :148.9   Min.   : 37.80   f:100   Row     :37
##  1st Qu.: 54.67   1st Qu.:174.0   1st Qu.: 66.53   m:102   T_400m  :29
##  Median : 63.03   Median :179.7   Median : 74.40           B_Ball  :25
##  Mean   : 64.87   Mean   :180.1   Mean   : 75.01           Netball:23
##  3rd Qu.: 74.75   3rd Qu.:186.2   3rd Qu.: 84.12           Swim    :22
##  Max.   :106.00   Max.   :209.4   Max.   :123.20           Field   :19
##                                                            (Other):47
```

```r
# (b)
table(ais$sex,ais$sport)
```

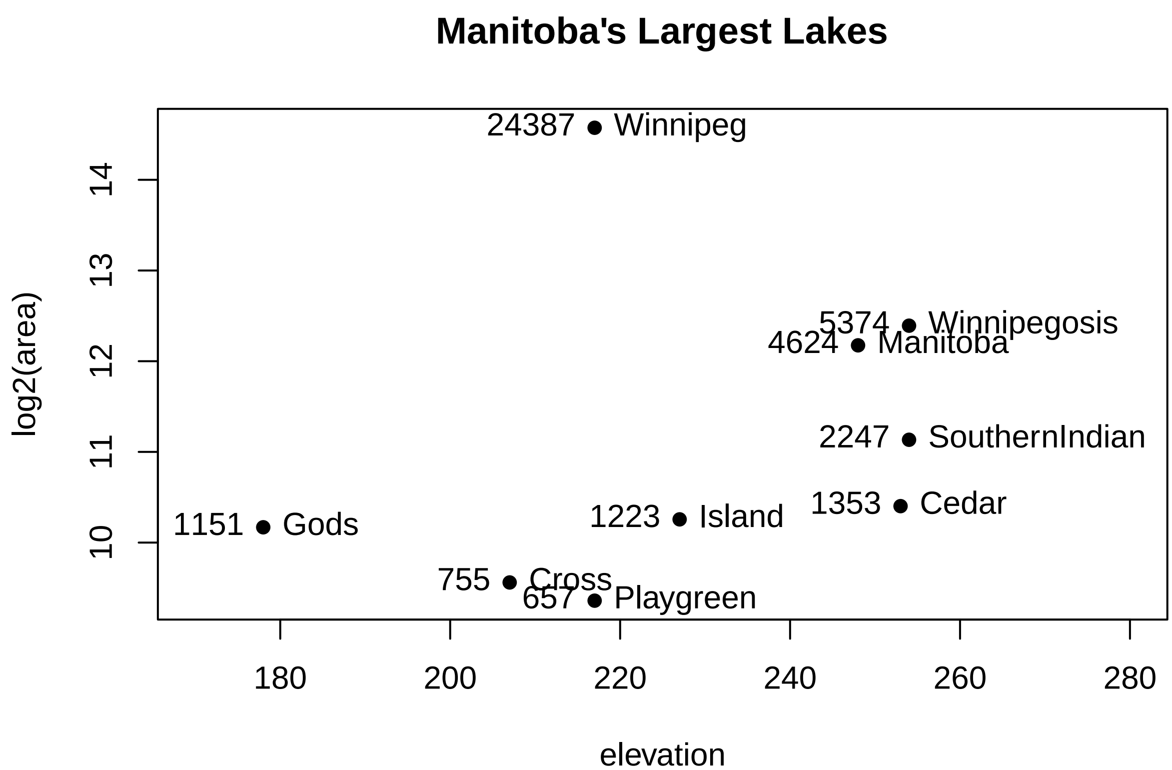
```
##
##     B_Ball Field Gym Netball Row Swim T_400m T_Sprnt Tennis W_Polo
##   f     13     7   4      23  22    9     11       4      7      0
##   m     12    12   0       0  15   13     18      11      4     17
```

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.

	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) 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):

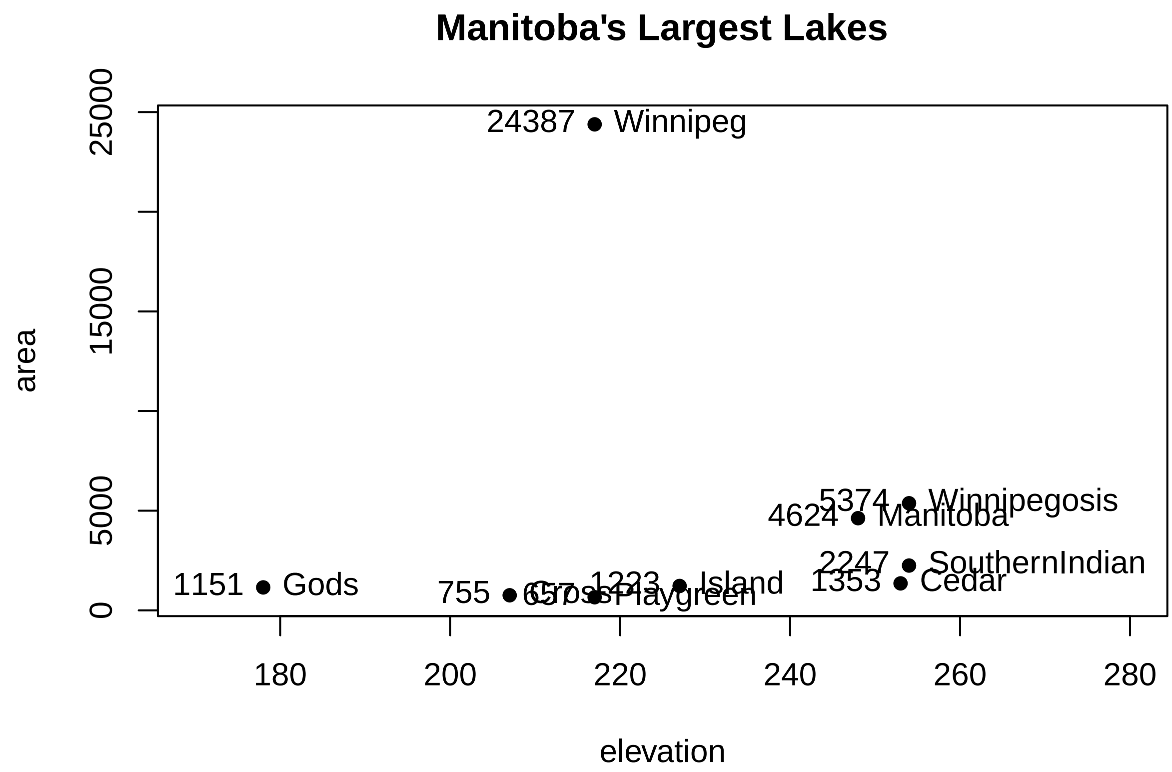
```
attach(Manitoba.lakes)
plot(log2(area) ~ elevation, pch=16, xlim=c(170,280))
# NB: Doubling the area increases log2(area) by 1.0
text(log2(area) ~ elevation, labels=row.names(Manitoba.lakes), pos=4)
text(log2(area) ~ elevation, labels=area, pos=2)
title("Manitoba's Largest Lakes")
```



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")
```



Ch1.6

```
Manitoba.lakes <- data.frame(
  elevation = c(217, 254, 248, 254, 253, 227, 178, 207, 217),
  area = c(24387, 5374, 4624, 2247, 1353, 1223, 1151, 755, 657)
)
row.names(Manitoba.lakes) <- c("Winnipeg", "Winnipegosis", "Manitoba", "SouthernIndian", "Cedar", "Isla
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
```

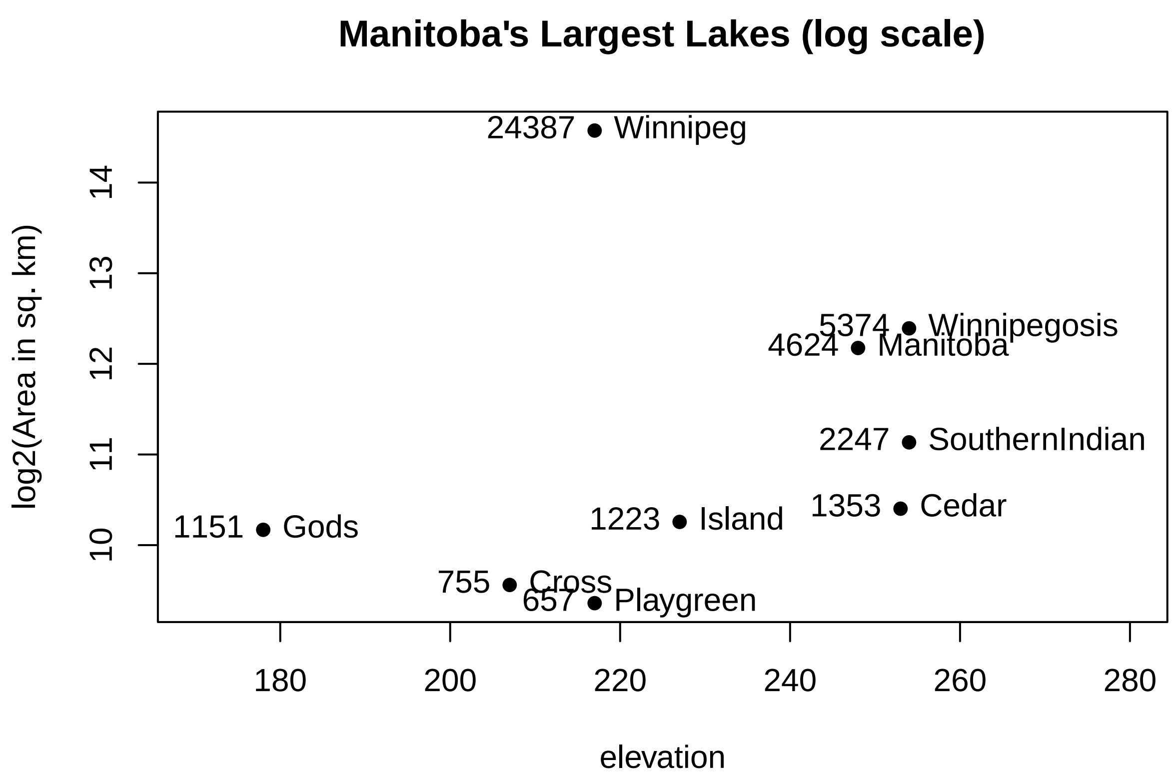
Ch1.6(a)

```
attach(Manitoba.lakes)
```

```
## The following objects are masked from Manitoba.lakes (pos = 3):
##
```

```
##     area, elevation
```

```
plot(log2(area) ~ elevation, pch=16, xlim=c(170,280),
     ylab = "log2(Area in sq. km)")
# 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 (log scale)")
```



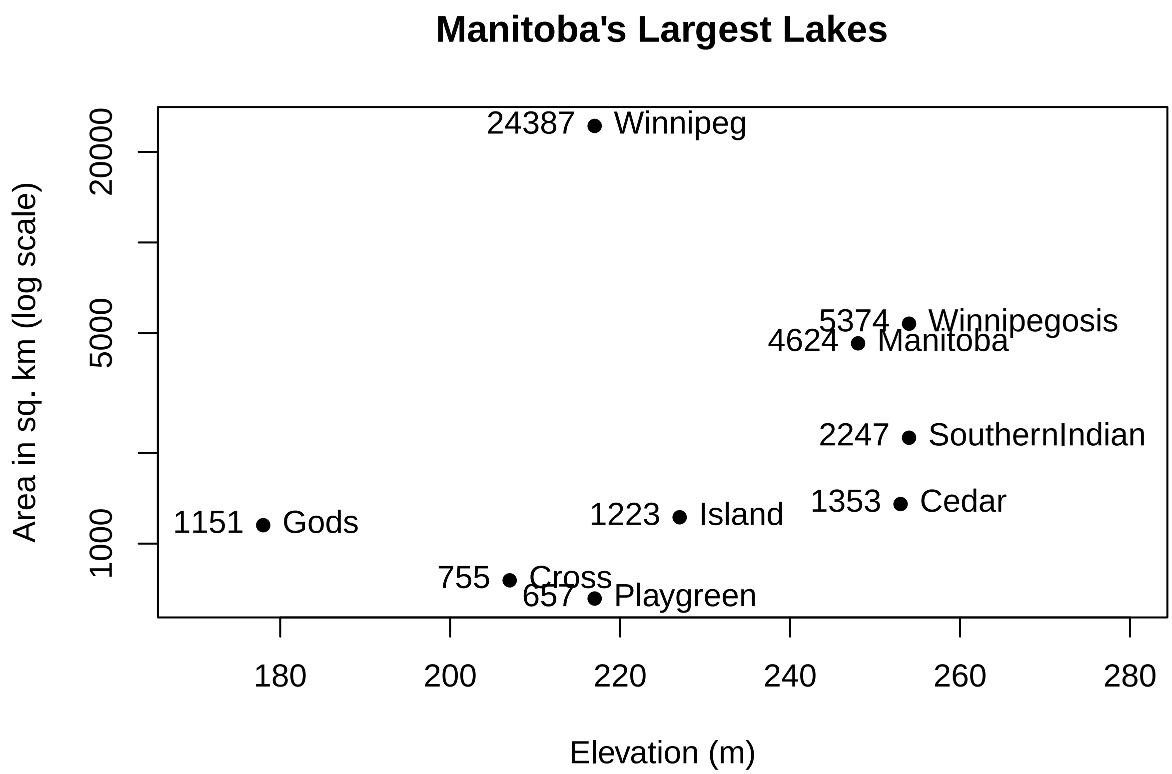

```
detach(Manitoba.lakes)
```

- Point label description: Each point in the map represents a lake. The text to the right of the point is the name of the lake, and the number to the left is the actual area of the lake (in square kilometers).
- Y-axis description: The Y-axis represents the base-2 logarithm of the lake area (`log2(area)`). On this scale, every increase of 1.0 on the Y-axis represents a doubling of the actual area of the lake. This logarithmic transformation helps to clearly show lakes with very different areas on the same map. ## Ch1.6(b)

```
attach(Manitoba.lakes)
```

```
## The following objects are masked from Manitoba.lakes (pos = 3):
##
##     area, elevation
```

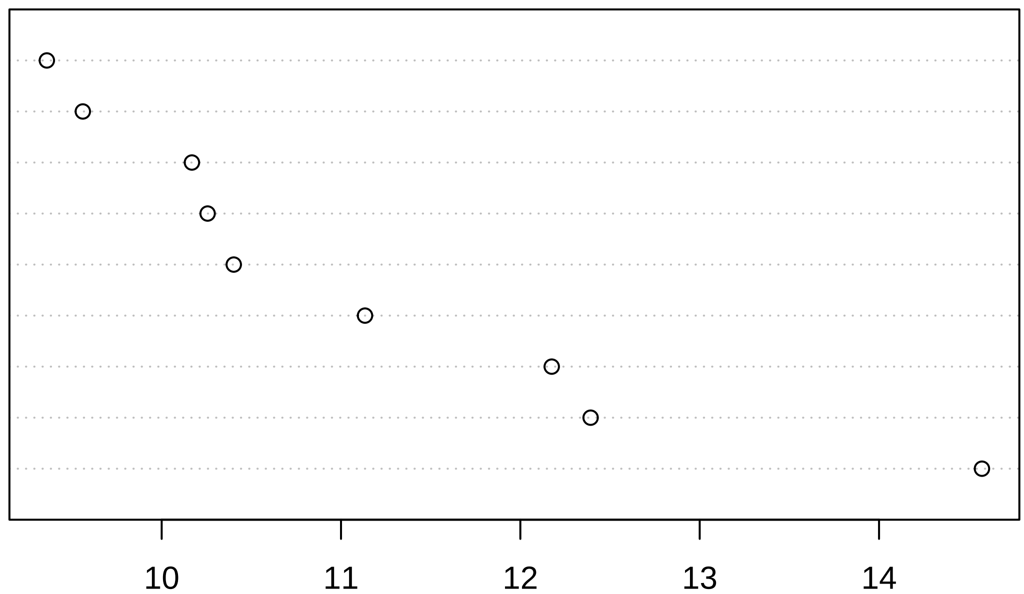
```
plot(area ~ elevation, pch = 16, xlim = c(170, 280), log = "y",
     ylab = "Area in sq. km (log scale)", xlab = "Elevation (m)")
text(area ~ elevation, labels = row.names(Manitoba.lakes), pos = 4)
text(area ~ elevation, labels = area, pos = 2)
title("Manitoba's Largest Lakes")
```




```
detach(Manitoba.lakes)
```

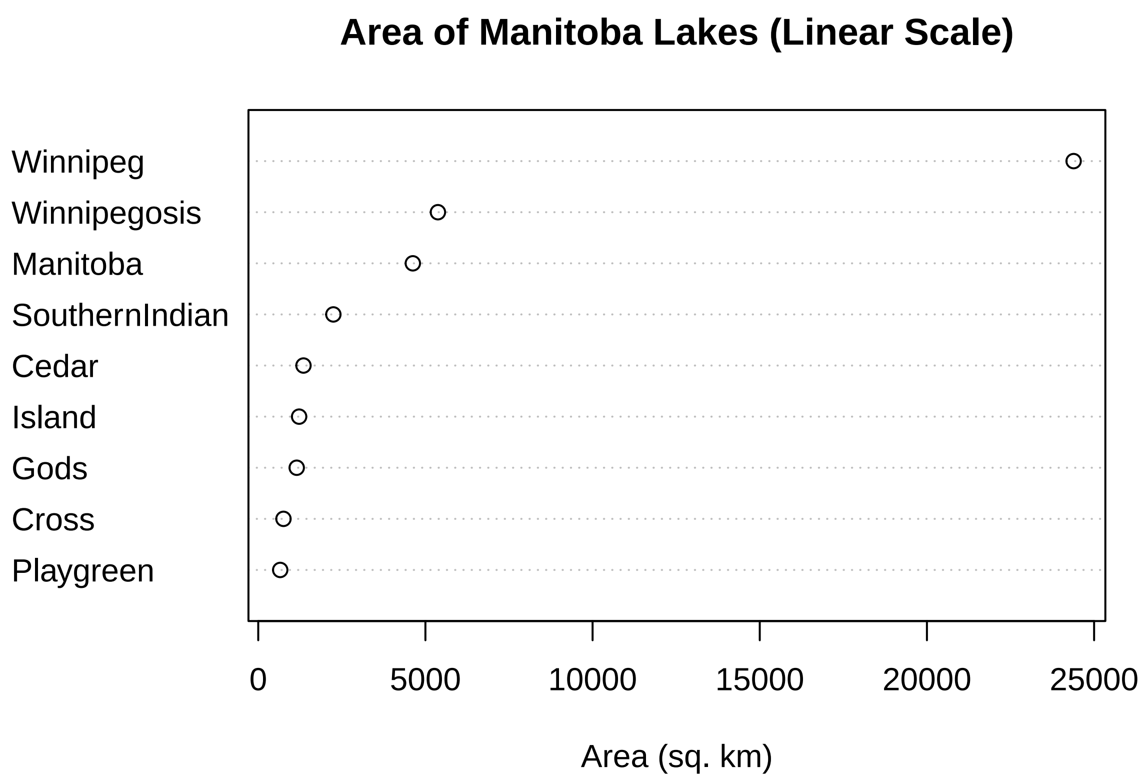
MB.Ch1.7. Look up the help page for the R function `dotchart()`. Use this function to display the areas of the Manitoba lakes (a) on a linear scale, and (b) on a logarithmic scale. Add, in each case, suitable labeling information.

```
dotchart(log2(area))
```

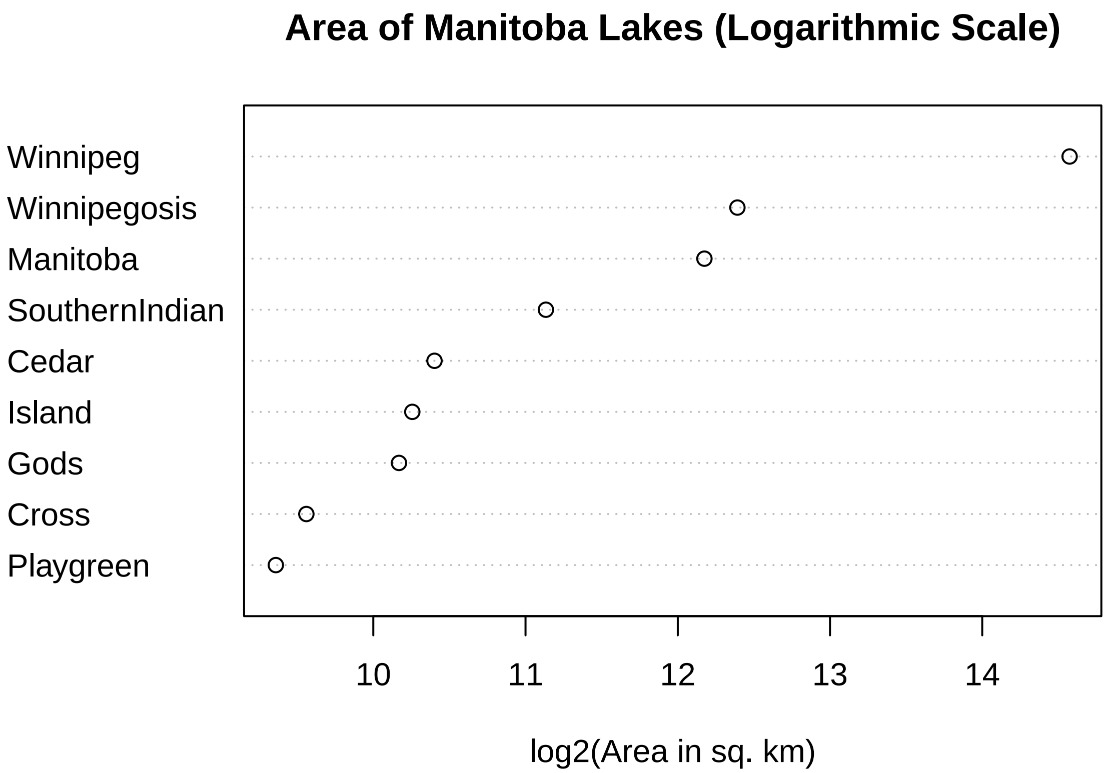


Ch1.7

```
Manitoba.lakes_sorted <- Manitoba.lakes[order(Manitoba.lakes$area), ]
# (a)
dotchart(Manitoba.lakes_sorted$area, labels = row.names(Manitoba.lakes_sorted),
         main = "Area of Manitoba Lakes (Linear Scale)",
         xlab = "Area (sq. km)")
```




```
# (b)
dotchart(log2(Manitoba.lakes_sorted$area), labels = row.names(Manitoba.lakes_sorted),
         main = "Area of Manitoba Lakes (Logarithmic Scale)",
         xlab = "log2(Area in sq. km)")
```



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

Ch1.8

```
total_area <- sum(Manitoba.lakes$area)
total_area
```

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