# Lab 1

## a) Variables and Types

### 1:

There is no difference

### 2:

logical, integer, real, complex, string,

### 3:

```
4 + T
```

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

```
#4 + "false"
```

### 4:

sum is not int the boolean format because '+' is an operation for numeric data types.

```
sum <- T + F
sum
```

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

```
typeof(sum)
```

```
## [1] "integer"
```

```
sum <- as.logical(sum)
typeof(sum)
```

```
## [1] "logical"
```

```
sum
```

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

## b) Data structures

### 5:

Vector, list, matrix, data frame, factors

## c) Vectors

### 6:

```
vec <- c(1, 2, 3)
vec
```

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

### 7:

We can't put variables of different types in a vector, since a vector is a random variable, so all variables in a vector must have the same type.

### 8:

```
feature_1 <- c(50 : 250)
feature_1
```

```
##   [1]  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67
##  [19]  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [37]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103
##  [55] 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
##  [73] 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139
##  [91] 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157
## [109] 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
## [127] 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193
## [145] 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211
## [163] 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229
## [181] 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247
## [199] 248 249 250
```

### 9:

```
length(feature_1)
```

```
## [1] 201
```

### 10:

```
mean(feature_1)
```

```
## [1] 150
```

```
var(feature_1)
```

```
## [1] 3383.5
```

```
sd(feature_1)
```

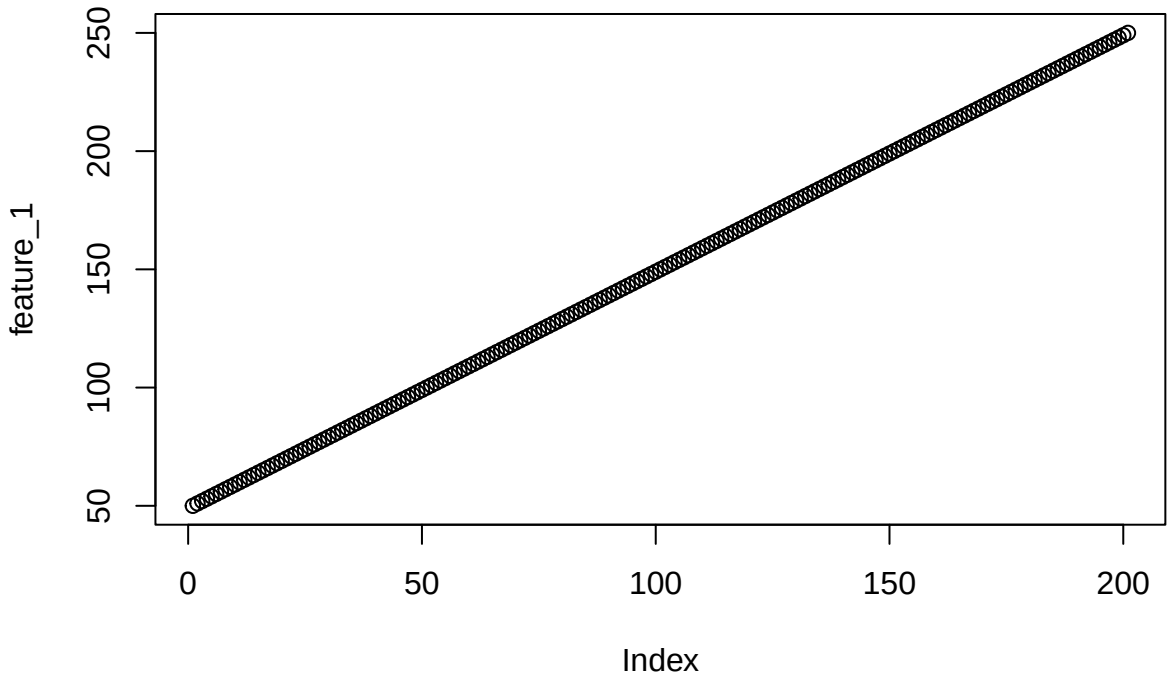
```
## [1] 58.16786
```

```
median(feature_1)
```

```
## [1] 150
```

**11:**

```
plot(feature_1)
```




**12:**

```
a <- feature_1[6 : 22]
a
```

```
##  [1] 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71
```

**13:**

```
b <- feature_1[c(6, 13, 21, 22, 43)]
b
```

```
## [1] 55 62 70 71 92
```

**14:**

```
c <- c(a, b)
c
```

```
##  [1] 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 55 62 70 71 92
```

### 15:

```
find_a <- function(str){
  return (grepl("a", str))
}

str <- c("joey", "phoebe", "monica", "chandler", "ross", "rachel")
sum(as.integer(lapply(str, find_a)))
```

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

## d) Factors

### 16:

Factor in R is **a variable used to categorize and store the data, having a limited number of different values**.

Factors have limited number of different values, while vectors don't have any limitation.

```
directions <- c("West", "East", "East", "North", "West", "West")
feature_2 <- factor (directions)
```

### 17:

It isn't possible, because 'South' isn't one of feature_2's levels.

### 18:

```
levels(feature_2) <- c(levels(feature_2), "South")
feature_2
```

```
## [1] West  East  East  North West  West
## Levels: East North West South
```

### 19:

There is nothing important to report, everything works well.

```
feature_2[1] <- "South"
feature_2
```

```
## [1] South East  East  North West  West
## Levels: East North West South
```

## e) Missing values

### 20:

NA

### 21:

```
feature_1[1] <- NA
feature_1
```

```
##   [1]  NA  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67
##  [19]  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [37]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103
##  [55] 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
##  [73] 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139
##  [91] 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157
## [109] 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
## [127] 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193
## [145] 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211
## [163] 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229
## [181] 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247
## [199] 248 249 250
```

### 22:

mean() function can't calculate feature_1's mean and returns NaN because first element of this vector is missing.

We can set "na.rm" argument, true in mean() function to resolve this problem.

```
mean(feature_1, na.rm = T)
```

```
## [1] 150.5
```

### 23:

```
feature_1[43] <- NA
which(is.na(feature_1))
```

```
## [1]  1 43
```

### 24:

**NULL represents the null object in R**. NULL is used mainly to represent the lists with zero length, and is often returned by expressions and functions whose value is undefined.

## f) Lists

### 25:

List is a list of variables which types could be different.

### 26:

list [x] will return a list which only contains the x_th element of list, while list [[x]] will return x_th element of list, it self.

```
lst <- list(4, 5, 6, list(7, 8, "xyz"))
```

## g) Naming

```
named_list <- list(a = 1, b = 2, c = 3, d = c(4, 5, 6, 7))
```

### 27:

First one returns a list contains 2, while other ones returns 2 itself.

it is a syntax sugar for [[]] because they both do the same thing.

```
named_list["b"]
```

```
## $b
## [1] 2
```

```
named_list[["b"]]
```

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

```
named_list$b
```

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

## h) Data Frames

### 28:

```
ncol(Orange)
```

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

### 29:

```
nrow(Orange)
```

```
## [1] 35
```

### 30:

```
f3 <- Orange$circumference
f3_tmp <- Orange[[3]]
```

### 31:

f3 is a vector.

Due to the dipersion of circumference feature, this data set had been chosen randomly.

```
table(f3)
```

```
## f3
##  30  32  33  49  51  58  62  69  75  81  87 108 111 112 115 120 125 139 140 142
##   3   1   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1   1   1   2
## 145 156 167 172 174 177 179 203 209 214
##   1   1   1   1   1   1   1   2   1   1
```

### 32:

Tree feature is catagorical

```
str(Orange)
```

```
## Classes 'nfnGroupedData', 'nfGroupedData', 'groupedData' and 'data.frame':   35 obs. of  3 variables
##  $ Tree         : Ord.factor w/ 5 levels "3"<"1"<"5"<"2"<..: 2 2 2 2 2 2 2 4 4 4 ...
##  $ age          : num  118 484 664 1004 1231 ...
##  $ circumference: num  30 58 87 115 120 142 145 33 69 111 ...
##  - attr(*, "formula")=Class 'formula'  language circumference ~ age | Tree
##   .. ..- attr(*, ".Environment")=<environment: R_EmptyEnv>
##  - attr(*, "labels")=List of 2
##   ..$ x: chr "Time since December 31, 1968"
##   ..$ y: chr "Trunk circumference"
##  - attr(*, "units")=List of 2
##   ..$ x: chr "(days)"
##   ..$ y: chr "(mm)"
```

### 33:

```
s29 <- Orange[29, ]
s29
```

```
##    Tree age circumference
## 29    5 118            30
```

### 34:

It is a data frame with one row.

### 35:

```
Orange[c(Orange$Tree == 3), ]
```

```
##    Tree  age circumference
## 15    3  118            30
## 16    3  484            51
## 17    3  664            75
## 18    3 1004           108
## 19    3 1231           115
## 20    3 1372           139
## 21    3 1582           140
```

### 36:

```
tmp <- Orange[1 : 10, c("Tree", "age")]
tmp
```

```
##    Tree  age
## 1     1  118
## 2     1  484
## 3     1  664
## 4     1 1004
## 5     1 1231
## 6     1 1372
## 7     1 1582
## 8     2  118
## 9     2  484
## 10    2  664
```

### 37:

```
median(Orange$age)
```

```
## [1] 1004
```

## i) Export and Import

### 38:

```
df_1 <- tail(Orange, n = 15)
```

### 39:

```
write.csv(df_1, "df_1.csv", row.names = F)
```

### 40:

```
df_2 <- read.csv("df_1.csv")
df_2
```

```
##    Tree  age circumference
## 1     3 1582           140
## 2     4  118            32
## 3     4  484            62
## 4     4  664           112
## 5     4 1004           167
## 6     4 1231           179
## 7     4 1372           209
## 8     4 1582           214
## 9     5  118            30
## 10    5  484            49
## 11    5  664            81
## 12    5 1004           125
## 13    5 1231           142
## 14    5 1372           174
## 15    5 1582           177
```