exercise#1

2024-03-07

A.Load the built-in warpbreaks dataset.

```
data("warpbreaks")
warpbreaks
```

```
##    breaks wool tension
## 1      26    A       L
## 2      30    A       L
## 3      54    A       L
## 4      25    A       L
## 5      70    A       L
## 6      52    A       L
## 7      51    A       L
## 8      26    A       L
## 9      67    A       L
## 10     18    A       M
## 11     21    A       M
## 12     29    A       M
## 13     17    A       M
## 14     12    A       M
## 15     18    A       M
## 16     35    A       M
## 17     30    A       M
## 18     36    A       M
## 19     36    A       H
## 20     21    A       H
## 21     24    A       H
## 22     18    A       H
## 23     10    A       H
## 24     43    A       H
## 25     28    A       H
## 26     15    A       H
## 27     26    A       H
## 28     27    B       L
## 29     14    B       L
## 30     29    B       L
## 31     19    B       L
## 32     29    B       L
## 33     31    B       L
## 34     41    B       L
## 35     20    B       L
## 36     44    B       L
## 37     42    B       M
## 38     26    B       M
## 39     19    B       M
## 40     16    B       M
```

```
## 41     39    B       M
## 42     28    B       M
## 43     21    B       M
## 44     39    B       M
## 45     29    B       M
## 46     20    B       H
## 47     21    B       H
## 48     24    B       H
## 49     17    B       H
## 50     13    B       H
## 51     15    B       H
## 52     15    B       H
## 53     16    B       H
## 54     28    B       H
```

A1.Find out, in a single command, which columns of warpbreaks are either numeric or integer. What are the data types of each column?

```
str(warpbreaks)
```

```
## 'data.frame':    54 obs. of  3 variables:
##  $ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
##  $ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
##  $ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 1 2 ...
```

A2.How many observations does it have?

```
#it has 54 observations.
```

A3.Is numeric a natural data type for the columns which are stored as such? Convert to integer when necessary.

```
typeof(warpbreaks$breaks)
```

```
## [1] "double"
```

```
typeof(warpbreaks$wool)
```

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

```
typeof(warpbreaks$tension)
```

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

```
#the data type of breaks, wool, and tension is double, integer, and integer.
```

A4.Error messages in R sometimes report the underlying type of an object rather than the user-level class. Derive from the following code and error message what the underlying type. Explain what is the error all about. Do not just copy the error message that was displayed.

B.Load the exampleFile.txt B1.Read the complete file using readLines.

```
file <- file("exampleFile.txt")

fileread <- readLines(file)
fileread
```

```
## [1] "// Survey data. Created : 21 May 2013"
## [2] "// Field 1: Gender"
## [3] "// Field 2: Age (in years)"
## [4] "// Field 3: Weight (in kg)"
```

```
## [5] "M;28;81.3"
## [6] "male;45;"
## [7] "Female;17;57,2"
## [8] "fem.;64;62.8"
```

B2.Separate the vector of lines into a vector containing comments and a vector containing the data. Hint: use grepl.

```
comments <- fileread[grepl("^//", fileread)]
comments
```

```
## [1] "// Survey data. Created : 21 May 2013"
## [2] "// Field 1: Gender"
## [3] "// Field 2: Age (in years)"
## [4] "// Field 3: Weight (in kg)"
```

```
datavector <- fileread[!grepl("^//", fileread)]
datavector
```

```
## [1] "M;28;81.3"      "male;45;"       "Female;17;57,2" "fem.;64;62.8"
```

B3.Extract the date from the first comment line and display on the screen "It was created data."

```
subcomments <- comments[1]
date <- gsub("// Survey data. Created : ", "", subcomments)
date
```

```
## [1] "21 May 2013"
```

```
cat("It was created, ", date)
```

```
## It was created,  21 May 2013
```

B4. B4a.Split the character vectors in the vector containing data lines by semicolon (;) using strsplit.

```
vectorsplit <- (strsplit(datavector, ";"))
vectorsplit
```

```
## [[1]]
## [1] "M"    "28"   "81.3"
##
## [[2]]
## [1] "male" "45"
##
## [[3]]
## [1] "Female" "17"     "57,2"
##
## [[4]]
## [1] "fem." "64"   "62.8"
```

B4b.Find the maximum number of fields retrieved by split. Append rows that are shorter with NA's.

```
maxvector <- max(lengths(vectorsplit))
maxvector
```

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

```
rowappend <- lapply(vectorsplit, function(x) c(x, rep(NA, maxvector - length(x))))
rowappend
```

```
## [[1]]
## [1] "M"    "28"   "81.3"
```

```
##
## [[2]]
## [1] "male" "45"   NA
##
## [[3]]
## [1] "Female" "17"     "57,2"
##
## [[4]]
## [1] "fem." "64"   "62.8"
```

B4c.Use unlist and matrix to transform the data to row-column format.

```
unlistdata <- unlist(rowappend)
unlistdata
```

```
##  [1] "M"      "28"     "81.3"   "male"   "45"     NA       "Female" "17"
##  [9] "57,2"   "fem."   "64"     "62.8"
```

```
datamatrix <- matrix(unlistdata, ncol = 4, nrow=3,
                     dimnames = list(c("row1","row2", "row3")))
datamatrix
```

```
##      [,1]   [,2]   [,3]     [,4]
## row1 "M"    "male" "Female" "fem."
## row2 "28"   "45"   "17"     "64"
## row3 "81.3" NA     "57,2"   "62.8"
```

B5d.From comment lines 2-4, extract the names of the fields. Set these as colnames for the matrix you just created.

```
fieldnames <- comments[2:4]
extfieldname <- gsub("//", "", fieldnames)
extfieldname
```

```
## [1] " Field 1: Gender"         " Field 2: Age (in years)"
## [3] " Field 3: Weight (in kg)"
```

```
rownames(datamatrix) <- extfieldname
datamatrix
```

```
##                         [,1]   [,2]   [,3]     [,4]
##  Field 1: Gender        "M"    "male" "Female" "fem."
##  Field 2: Age (in years) "28"   "45"   "17"     "64"
##  Field 3: Weight (in kg) "81.3" NA     "57,2"   "62.8"
```