INFDEV03-7/INFDEV23-7   Practical Exam First Chance - 15 April 2019

Instructions:

- make sure to have R, RStudio and the ISwR library installed on your laptop, before the start of the practical exam
- put your laptop in flight mode (use the function key with the symbol of an airplane)
- exit all Internet browsers
- exit all mail, chat and any other messaging applications
- any sign to try to use any means of communication using your laptop will be registered as (attempted) fraud
- use of any other mobile devices is strictly forbidden
- you will be handed in a base-R cheat sheet that you are allowed to use.
- The final grade of this practical exam is calculated with the following formula: FinalGrade = 10*TotalPoints/MaxPoints

- Use R Studio to do your analysis and write the answers (R code fragments + short text answers) <u>on the exam form.</u>
- If the question asks to make a plot, you are expected to write down the R code to make that plot (you are not expected to draw the plot itself).
- For the questions that require you to test hypothesis, follow the steps:
    - Specify the two hypotheses of the test (null and alternative).
    - Write down the R-code to perform the test
    - Explain your decision (whether you accept or reject the null hypothesis) based on the R-output of your test.

Base R
Cheat Sheet

Getting Help

Accessing the help files

?mean
Get help of a particular function.

help.search('weighted mean')
Search the help files for a word or phrase.

help(package = 'dplyr')
Find help for a package.

More about an object

str(iris)
Get a summary of an object's structure.

class(iris)
Find the class an object belongs to.

Using Libraries

install.packages('dplyr')
Download and install a package from CRAN.

library(dplyr)
Load the package into the session, making all its functions available to use.

dplyr::select
Use a particular function from a package.

data(iris)
Load a build-in dataset into the environment.

Working Directory

getwd()
Find the current working directory (where inputs are found and outputs are sent).

setwd('C://file/path')
Change the current working directory.

Use projects in RStudio to set the working directory to the folder you are working in.

Vectors

Creating Vectors

c(2, 4, 6)	2 4 6	Join elements into a vector
2:6	2 3 4 5 6	An integer sequence
seq(2, 3, by=0.5)	2.0 2.5 3.0	A complex sequence
rep(1:2, times=3)	1 2 1 2 1 2	Repeat a vector
rep(1:2, each=3)	1 1 1 2 2 2	Repeat elements of a vector

Vector Functions

sort(x)
Return x sorted.

rev(x)
Return x reversed.

table(x)
See counts of values.

unique(x)
See unique values.

Selecting Vector Elements

By Position

x[4] — The fourth element.

x[-4] — All but the fourth.

x[2:4] — Elements two to four.

x[-(2:4)] — All elements except two to four.

x[c(1, 5)] — Elements one and five.

By Value

x[x == 10] — Elements which are equal to 10.

x[x < 0] — All elements less than zero.

x[x %in% c(1, 2, 5)] — Elements in the set 1, 2, 5.

Named Vectors

x['apple'] — Element with name 'apple'.

Programming

For Loop

```
for (variable in sequence){
    Do something
}
```

Example

```
for (i in 1:4){
    j <- i + 10
    print(j)
}
```

While Loop

```
while (condition){
    Do something
}
```

Example

```
while (i < 5){
    print(i)
    i <- i + 1
}
```

If Statements

```
if (condition){
    Do something
} else {
    Do something different
}
```

Example

```
if (i > 3){
    print('Yes')
} else {
    print('No')
}
```

Functions

```
function_name <- function(var){
    Do something
    return(new_variable)
}
```

Example

```
square <- function(x){
    squared <- x*x
    return(squared)
}
```

Reading and Writing Data

Input	Ouput	Description
df <- read.table('file.txt')	write.table(df, 'file.txt')	Read and write a delimited text file.
df <- read.csv('file.csv')	write.csv(df, 'file.csv')	Read and write a comma separated value file. This is a special case of read.table/write.table.
load('file.RData')	save(df, file = 'file.Rdata')	Read and write an R data file, a file type special for R.

Conditions								
	a == b	Are equal	a > b	Greater than	a >= b	Greater than or equal to	is.na(a)	Is missing
	a != b	Not equal	a < b	Less than	a <= b	Less than or equal to	is.null(a)	Is null

Types

Converting between common data types in R. Can always go from a higher value in the table to a lower value.

as.logical	TRUE, FALSE, TRUE	Boolean values (TRUE or FALSE).
as.numeric	1, 0, 1	Integers or floating point numbers.
as.character	'1', '0', '1'	Character strings. Generally preferred to factors.
as.factor	'1', '0', '1', levels: '1', '0'	Character strings with preset levels. Needed for some statistical models.

Maths Functions

`log(x)`	Natural log.	`sum(x)`	Sum.
`exp(x)`	Exponential.	`mean(x)`	Mean.
`max(x)`	Largest element.	`median(x)`	Median.
`min(x)`	Smallest element.	`quantile(x)`	Percentage quantiles.
`round(x, n)`	Round to n decimal places.	`rank(x)`	Rank of elements.
`sig.fig(x, n)`	Round to n significant figures.	`var(x)`	The variance.
`cor(x, y)`	Correlation.	`sd(x)`	The standard deviation.

Variable Assignment

```
> a <- 'apple'
> a
[1] 'apple'
```

The Environment

`ls()`	List all variables in the environment.
`rm(x)`	Remove x from the environment.
`rm(list = ls())`	Remove all variables from the environment.

You can use the environment panel in Studio to browse variables in your environment.

Matrixes

```
m <- matrix(x, nrow = 3, ncol = 3)
```
Create a matrix from x.

`m[2, ]` - Select a row

`m[ , 1]` - Select a column

`m[2, 3]` - Select an element

`t(m)` Transpose

`m %*% n` Matrix Multiplication

`solve(m, n)` Find x in: m * x = n

Lists

```
l <- list(x = 1:5, y = c('a', 'b'))
```
A list is collection of elements which can be of different types.

`l[[2]]` Second element of l.

`l[1]` New list with only the first element.

`l$x` Element named x.
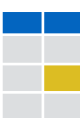
`l['y']` New list with only element named y.

Data Frames

Also see the **dplyr** library.

```
df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))
```
A special case of a list where all elements are the same length.

x	y
1	a
2	b
3	c

List subsetting

`df$x`   `df[[2]]`

Understanding a data frame

`View(df)` See the full data frame.

`head(df)` See the first 6 rows.

Matrix subsetting

`df[ , 2]`

`df[2, ]`

`df[2, 2]`

`nrow(df)` Number of rows.

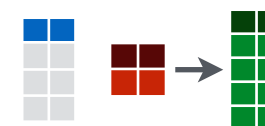`ncol(df)` Number of columns.

`dim(df)` Number of columns and rows.

`cbind` - Bind columns.
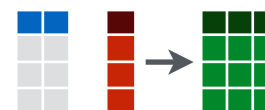
`rbind` - Bind rows.

Strings

Also see the **stringr** library.

`paste(x, y, sep = ' ')`	Join multiple vectors together.
`paste(x, collapse = ' ')`	Join elements of a vector together.
`grep(pattern, x)`	Find regular expression matches in x.
`gsub(pattern, replace, x)`	Replace matches in x with a string.
`toupper(x)`	Convert to uppercase.
`tolower(x)`	Convert to lowercase.
`nchar(x)`	Number of characters in a string.

Factors

`factor(x)` Turn a vector into a factor. Can set the levels of the factor and the order.

`cut(x, breaks = 4)` Turn a numeric vector into a factor but 'cutting' into sections.

Statistics

`lm(x ~ y, data=df)` Linear model.

`glm(x ~ y, data=df)` Generalised linear model.

`summary` Get more detailed information out a model.

`t.test(x, y)` Preform a t-test for difference between means.

`pairwise.t.test` Preform a t-test for paired data.

`prop.test` Test for a difference between proportions.

`aov` Analysis of variance.

Distributions

	Random Variates	Density Function	Cumulative Distribution	Quantile
Normal	`rnorm`	`dnorm`	`pnorm`	`qnorm`
Poison	`rpois`	`dpois`	`ppois`	`qpois`
Binomial	`rbinom`	`dbinom`	`pbinom`	`qbinom`
Uniform	`runif`	`dunif`	`punif`	`qunif`

Plotting

Also see the **ggplot2** library.

`plot(x)` Values of x in order.

`plot(x, y)` Values of x against y.

`hist(x)` Histogram of x.

Dates

See the **lubridate** library.