GENERAL COMMANDS	
Help (for the command)	?*command*
Comment	#
Summation, subtraction	+, -
Multiplication, ratio	*, /
Variable assignment	= or <-
Not equal	!=
Sinus, cosinus	sin(*argument*), cos(*argument*)
π	pi
Exponentiation (power)	^
Square root	sqrt(*argument*)
Root of n-th degree	*argument*^(1/n)
e	exp(1)
Logarithm	log(*argument*, base=*base of logarithm*)
Vector	c(*elements divided by commas*)
Indicating i-th element of a vector	*vector*[i]
Vector – sequence with the number of components equally spread one from the other	seq(*from, to,* length=*number of classes*)
Vector – sequence with steps	seq(*from, to,* by=*step*)
Vector of replicated subvector	rep(*vector*, times=*number of replications*)
Vector of replicated components of subvector	rep(*vector*, each=*number of replications*)
Sum of components in a vector	sum(*vector*)
Number of components in vector	length(*vector*)
Changing the order of vector components	rev(*vector*)
Removing the components from vector	*vector*[-c(*indices of components to remove*)]
Indicating components of vector smaller than "k"	*vector*[*vector*<k]
Indicating indices of vector components smaller than "k"	which(*vector*<k)
Indicating indices of minimal and maximal component	which.min(*vector*) which.max(*vector*)
Matrix from column vectors	cbind(x1, x2,…, xm)
Matrix from row vectors	rbind(x1, x2,…, xm)
Multiplication of matrices	%*%
Determinant	det(*matrix*)
Indicating (i,j)-th entry of a matrix	*matrix*[[i,j]]
Indicating i-th row / j-th column of a matrix	*matrix*[i,] *matrix*[,j]
Transposition	t(*matrix*)
Diagonal of matrix	diag(*matrix*)
Dimension of matrix	dim(*matrix*)
Inverse of matrix	solve(*matrix*)
Number of rows and columns of matrix	nrow(*matrix*), ncol(*matrix*)
Percentage notation („scales" package)	percent(*number*)
Division of window with graphs	par(mfrow=c(*n*,*m*))
Simple graph representing points (x,y) or data	plot(*x, y*) plot(*data*)
Graph representing a function of one variable	curve(*function of x, x lower bound, x upper bound*)
PACKAGES	
Package Installing	install.packages(„*name*")
Package loading	library(*name*)
„FOR", „IF", FUCTIONS AND PROCEDURES	
„for"	for (*variable* in *beginning*:*end*){*what to do*} Caution! Variable can be also a vector!
„if"	if (*condition*) {*what to do*} else {*what to do*}

functions/procedures	*name*= function (*arguments*){
	what to do
	return(*output*)}

DESCRIPTIVE STATISTICS	
Loading data	read.csv(„*name*”, sep=”;”)
Loading data in polish coding („,” used for decimal)	read.csv(„*name*”, sep=”;”, dec=”,”)
Loading data with labels	read.csv(„*name*”, sep=”;”, head=TRUE)
Creating data as a table of data (x1, x2,…, xm are column labels)	data.frame(x1, x2,…, xm)
Class of data	class(*data*)
Vector of labels	names(*data*)
Mean	mean(*data*)
Minimum and maximum	min(*data*), max(*data*)
Quartiles	quantile(*data*)
Quantiles (chosen)	quantile(*data*, probs=vector of probabilities)
Variance and standard deviation	var(*data*), sd(*data*)
Function from rows of matrix (table)	apply(*matrix,* 1, *function*)
Function from columns of matrix (table)	apply(*matrix,* 2, *function*)
Central tendency measures – all	summary(*data*)
Histogram for discrete data (line graph) Caution! „arm” package required	discrete.histogram(*data*)
Histogram (for grouped data)	hist(*data*, main=*title*, xlab=*label of x*)

Automatic declaration of graphics titles e.g. using “for” procedure (with space or without respectively)	paste(„*text*”, *name*)
	paste0(„*text*”, *name*)

Computing e.g. mean if the data have different lengths	mean(na.omit(*data*))
Grouping data into frequency table	table(*data*)
Frequency table with intervals	cut(*data*, breaks = *# of classes*)
Pie chart (of point frequency table)	pie(table(*data*))
Pie chart (of point frequency table)	pie(table(cut(*data*, breaks = *# of classes*)))
Box plot	boxplot(*data*)
RANDOM VARIABLES AND DISTRIBUTIONS	
Probability/density function (d – density)	d*name*
CDF (p – probability)	p*name*
Quantile (q – quantile)	q*name*
Random generating (r – random)	r*name*
Distribution names	
binomial	binom
Poisson	pois
exponential	exp
normal	norm
t-Student	t
chi-square	chisq
F Snedecor	f
Line graph of discrete distribution	plot(x, d*name*(x, *parameters*))
Drawing density function	curve(d*name*(x, *parameters*))

CONFIDENCE INTERVALS (CI) AND HYPOTHESES TESTING	
CI for μ under normality with known σ Caution! „BSDA” package required	z.test(*data*, sigma.x=σ, conf.level=1-α)
CI for μ – large sample Caution! „BSDA” package required	zsum.test(*sample mean, (sample) stand. dev., sample size*, conf.level=1-α)
CI for μ under normality when σ is unknown	t.test(*data*, conf.level=1-α)
CI for σ^2 Caution! „TeachingDemos” package required	sigma.test(*data*, conf.level=1-α)
CI for probability of success (proportion) *p*	binom.test(*no. of successes, sample size*, conf.level=1-α)
Only CI as output	*NameOfTest*$conf.int
Hypothesis about μ under normality with known σ Caution! „BSDA” package required	z.test(*data*, sigma.x=σ, alternative=”two.sided”, mu=*tested mean*)
Hypothesis about μ when the sample is large Caution! „BSDA” package required	zsum.test(*sample mean, sample stand. dev., sample size*, alternative=”greater”, mu=*tested mean*)
Hypothesis about μ under normality with unknown σ	t.test(*data*, alternative=”less”, mu=*tested mean*)
Hypothesis about σ^2 Caution! „TeachingDemos” package required	sigma.test(*data*, sigma=*tested sigma*, alternative=”two.sided”)
Hypothesis about probability of success (proportion) *p*	binom.test(*no. of successes, sample size*, p=*tested probability*, alternative=”two.sided”)
Only p-value as an output	*NameOfTest*$p.value

COMPARISON OF TWO POPULATIONS	
CI for difference of means under normality (equal population variances)	t.test(*data1*, *data2*, var.equal=TRUE, conf.level= 1-α)
CI for difference of means under normality (unequal population variances)	t.test(*data1*, *data2*, var.equal=FALSE, conf.level= 1-α)
CI for difference of means (large samples) Caution! “BSDA” package required	zsum.test(*sample mean 1*, *stand. dev.* 1, *sample size* 1, *sample mean 2*, *stand. dev.* 2, *sample size* 2, conf.level=1-α)
CI for the ratio of variances Caution! „PairedData” package required	var.test(*data1*, *data2*, conf.level=1-α)
CI for the difference of proportions	prop.test(c(*T1*,*T2*), c(*n1*,*n2*), conf.level=1-α)
Hypothesis about difference of means (equal population variances)	t.test(*data1*, *data2*, mu=*tested difference of means*, var.equal=TRUE, alternative=”two.sided”)
Hypothesis about difference of means (unequal population variances)	t.test(*data1*, *data2*, mu=tested difference of means, var.equal=FALSE, alternative=”two.sided”)
Hypothesis about difference of means (large samples) Caution! “BSDA” package required	zsum.test(*sample mean 1*, *stand. dev.* 1, *sample size* 1, *sample mean 2*, *stand. dev.* 2, *sample size* 2, mu=*tested difference of means*, alternative=”two.sided”)
Hypothesis about ratio of variances Caution! „PairedData” package required	var.test(*data1*, *data2*, ratio=*tested ratio of variances*, alternative=”two.sided”)
Hypothesis about equality of proportions	prop.test(c(*T1*,*T2*), c(*n1*,*n2*), alternative=”two.sided”)

ANALYSIS OF VARIANCE (ANOVA)	
CAUTION! To perform ANOVA data has to be arrange in a proper way! data.frame(*measurements*, *methods*)	
Bartlett test of homogeneity of variances	bartlett.test(*measurements~treatments*)
Analysis of variance	anova(lm(*measurements~treatments*))
TukeyTest HSD of homogeneous treatments	TukeyHSD(aov(*measurements~treatments*),ordered=TRUE)
Drawing simultaneous confidence intervals	plot(TukeyHSD(aov(*measurements~treatments*),ordered=TRUE))
REGRESSION ANALYSIS	
Covariance	cov(*data1*, *data2*)
Correlation	cor(*data1*, *data2*)
Point graph of bivariate relation	plot(*x*, *y*)
Regression line	lm(*y~x*)
Hypothesis about significance of regression	anova(lm(*y~x*)) lub summary(lm(*y~x*))
Point graph and regression line together	plot(*x*, *y*); abline(*regression line*)
Prediction of missing values	predict(*regression line*, data.frame(c(*x1*, …, *xk*)))
CHI-SQUARE TESTS	
Goodness of fit test (qualitative data)	chisq.test(*observed frequencies*, p=*expected probabilities*)

Normality tests
Caution! “nortest” package required
	pearson.test(*data*, adjusted=T) pearson.test(*data*, adjusted=F)
	lillie.test(*data*)
	shapiro.test(*data*)

Test of independence of two variables	chisq.test(data.frame(*data1*, *data2*))