

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <Guano Anderson >
31
32
33 ######
34 # 1: R basics      (13 points)          #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1<-seq(10,1,-1)
39 v1
40 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
41 ... not use the c() command.
42 v2<-seq(3,30,3)
43 v2
44 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
45
46 v3<-seq(1,291,10)
47 v4<-seq(2,292,10)
48 v5<-seq(3,293,10)
49 v6<-seq(4,294,10)
```

```
48 v7<-seq(10,300,10)
49
50 M<-rbind(v3,v4,v5,v6,v7)
51 M
52 # 1.4 Print the first row of M
53 M[1,]
54 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
# regardless of the dimension of v1.
55 tail(v1,1)
56 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
57 (12/(19-7))^(1/5)
58
59 (log10(1)+log10(2))/((pi-1)/(pi-1))
60
61 log10(sin(2)/exp(2))
62 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
63 u<-10^5
64 # 1.8 If you would now run the line
65 print(U)
66 # you would get an error message. Which important concept of R is the
... reason
67 # for this error? Answer in 1 sentence as a comment.
68
69
70 ## R is case sensitive, which means that u and V are two different
... variables
71
72 #####
73 # 2: Data and Logical conditions (20 points) #
74 #####
75 # The file "Eudata.csv" contains data about the (still 28) EU countries.
76 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
77 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
78
79 # The following line loads the data into an R dataframe
80 # Hint: Use Session/Set Working Directory/To Source File Location
81 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
82
83 head(Eudata)
84 # 2.1 How many countries are there in the dataset?
85 nrow(Eudata)
86
87 # 2.2 Calculate the total population of the EU
88 sum(Eudata$Population)
89 # 2.3 Print the population of the smallest and largest EU country by Area
90 Eudata$Population[which.min(Eudata$Area)]# smallest
91
92 Eudata$Population[which.max(Eudata$Area)]#Largest
93
94 # 2.4 Calculate the number of countries that are members of the Eurozone
```

```
95 sum(Eudata$Eurozone==1)
96 # 2.5 Calculate the total GDP of all Eurozone members
97
98 sum(Eudata$GDP[Eudata$Eurozone==1])
99
100 # 2.6 Calculate the GDP per capita in euros
101 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
102 EU
103 ...
104 #a
105 gdp_pc_Eu<-sum((Eudata$GDP)*1e6)/sum(Eudata$Population)
106 gdp_pc_Eu
107 #b
108 gdp_pc_EZ<-(sum(Eudata$GDP[Eudata$Eurozone==1])*1e6)/sum(Eudata$Population[Eudata$Eurozone==1])
109 gdp_pc_EZ
110 #c
111 gdp_pc_no_Eu_memb<-(sum(Eudata$GDP[Eudata$Eurozone==0])*1e6)/sum(Eudata$Population[Eudata$Eurozone==0])
112 gdp_pc_no_Eu_memb
113 # 2.7 When was the EU founded?
114 #      Hint: this must be the earliest year in which any country became a member
115 min(Eudata$Accession)
116 # 2.8 Calculate the number of EU founding members
117 sum(Eudata$Accession==1953)
118 # 2.9 Only now you discover that the data set still contains the UK.
119 #      Permanently remove the UK from the dataframe "Eudata"
120 Eudata<-subset(Eudata,Code != "GB")
121 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st January 2026.
122 #      Permanently update the dataframe "Eudata" accordingly.
123
124
125
126 #####
127 # 3: Simulation and probability (15 points) #
128 #####
129 # 3.1 Use R to produce one roll of a dice.
130 One_roll_dice<-sample(1:6,1,T)
131 One_roll_dice
132 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
133 k<-sample(1:6,1000,T)
134 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
135 #      a "4" or "5" from a dice
136 mean(k==4 | k==5)
137 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of a dice
138 ...
139 #      Using "k" and "m" from (3.2) and (3.4), estimate the expected value
140 #      and variance of the random variable z = 2k-m
141 m<-sample(1:6,1000,T)
```

```
141 z<-2*k-m
142 z
143 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
144 #      a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
145 stock<-rnorm(100,0.12,0.2)
146 stock
147 # 3.6 What is the probability of a negative stock return?
148 #      Answer this question ...
149 #      (a) by using the variable "stock" from (3.5)
150 mean(stock<0)
151
152 #      (b) by calculating the (theoretical) probability for a normal
... distribution
153 pnorm(0,mean = 0.12,sd=0.2)
154
155 ######
156 # 4: Functions and Optimization      (25 points)#
157 #####
158 # 4.1 Create the function f(x)=x^2 in R
159
160 f<-function(x){
161   x^2
162 }
163 # 4.2 Calculate the value of f for x=1
164 f(1)
165 # 4.3 Create a plot of the function for the interval [-2, 2]
166 #      If in doubt, type "?plot" to get the help file for the function
167 curve(f,from = -2,to=2)
168
169 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
170 #      function. Store the result of your minimization (the location of
... the minimum)
171 #      in a variable called xmin
172 res<-optim(par=0,fn=f,method = "BFGS")
173 xmin<-res$par
174 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
175 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
176 #      Store the result of your minimization (the location of the minimum)
... in a
177 #      variable called xmin_grid
178
179 grid_search<-optim(par = 0, fn= f, gr = 100, method = "L-BFGS-B", lower =
... -2, upper = 2)
180 xmin_grid<-
181   optim()
182 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
```

```
183 #      are not identical. Why?  
184  
185  
186 #####  
187 # 5: Functions and algorithms (25 points) #  
188 #####  
189 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
... is valid. It goes  
190 #      like this  
191 #      a) process individual digits from right to left  
192 #      b) leave digits number 1,3,5 etc (counted from right) unchanged  
193 #      c) multiply digits 2,3,6 etc (counted from right) by 2  
194 #      d) if a digit (after multiplying by 2) is larger than 9, subtract  
... 9  
195 #      e) calculate the sum of all (processed) digits  
196 #      IF the result can be divided by 10, the number is a valid credit  
... card number  
197 #  
198 #      Example: 63487  
199 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do  
... not change 4,  
#                  multiply 3 by 2 and do not subtract anything, do  
... not change 6  
#                  7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is  
... valid  
202 #      Hint: The operation x %% y yields the remainder of the division  
... x/y.  
203 #      For instance, 7 %% 4 gives 3  
204  
205 # Write a function called checkLuhn that takes as argument a vector of  
... individual digits  
206 # and returns TRUE if the number is a valid number and FALSE if it is not  
... valid.  
207 # The following line takes a *valid* credit card number and creates a  
... vector with single digits.  
208 # You can use this to test your function  
209 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn  
... algorithm works from  
210 # right to left.  
211 digits <- as.numeric(unlist(strsplit("5019717010103742","")))  
212  
213 checkLuhn<-function(v1){  
214   v1<-c(6,3,4,8,7)  
215   value<-length(v1)  
216   index<-1  
217   somma<-0  
218   while (value>=1) {  
219     if(index%%2==0){  
220       v1[value]<-v1[value]*2  
221       if(v1[value]>9){  
222         v1[value]<-v1[value]-9  
223     }  
224   }
```

```
224     }
225     somma<-somma+v1[value]
226     value<-value-1
227     index<-index+1
228   }
229   if(somma%%10==0){
230     return(TRUE)
231   }else{
232     return(FALSE)
233   }
234 }
235 checkLuhn(digits)
236
237
238
239
240
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name (2 points) #
28 #####
29 # 0.1 Write your name below as a comment
30 # <your name>
31 #Anna Bonera
32
33 #####
34 # 1: R basics (13 points) #
35 #####
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 <- seq(10,1,-1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 <- seq(3,30,3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43
44 # 1.4 Print the first row of M
45
46 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
#     regardless of the dimension of v1.
tail(v1,1)
```

```
47 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.  
48 (12/(19-7))^(1/5)  
49 (log(1)+log(2))/((pi+1)/(pi-1))  
50 log((sin(2))/exp(2))  
51 # 1.7 Create a variable "u" with the value "ten to the power minus 5"  
52 u <- (10)^(-5)  
53 # 1.8 If you would now run the line  
54 # print(U)  
55 # you would get an error message. Which important concept of R is the  
... reason  
56 # for this error? Answer in 1 sentence as a comment.  
57  
58 because the name of variables are case sensitive, so U is not the same as  
... u  
59 #####  
60 # 2: Data and Logical conditions (20 points) #  
61 #####  
62 # The file "Eudata.csv" contains data about the (still 28) EU countries.  
63 # The columns are: County Name, Code, Capital, Accession (=Year of  
... membership),  
64 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if  
... member)  
65  
66 # The following line loads the data into an R dataframe  
67 # Hint: Use Session/Set Working Directory/To Source File Location  
68 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)  
69  
70 # 2.1 How many countries are there in the dataset?  
71  
72 # 2.2 Calculate the total population of the EU  
73 sum(Eudata$Population)  
74 # 2.3 Print the population of the smallest and largest EU country by Area  
75 sum(Eudata$Population[which.min(Eudata$Area)])  
76 sum((Eudata$Population[which.max(Eudata$Area)]))  
77 # 2.4 Calculate the number of countries that are members of the Eurozone  
78 sum(Eudata$Eurozone==TRUE)  
79 # 2.5 Calculate the total GDP of all Eurozone members  
80 sum(Eudata$GDP[Eudata$Eurozone==TRUE])  
81 # 2.6 Calculate the GDP per capita in euros  
82 # (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone  
... EU  
83 sum(Eudata$GDP/Eudata$Population)  
84 sum(Eudata$GDP[Eudata$Eurozone==TRUE]/Eudata$Population[Eudata$Eurozone==  
... TRUE])  
85 sum(Eudata$GDP[Eudata$Eurozone==FALSE]/Eudata$Population[Eudata$Eurozone=  
... =FALSE])  
86 # 2.7 When was the EU founded?  
87 # Hint: this must be the earliest year in which any country became a  
... member  
88  
89 # 2.8 Calculate the number of EU founding members  
90 EU_founding_members= sum(Eudata$Accession==1953)
```

```
91
92 # 2.9 Only now you discover that the data set still contains the UK.
93 #     Permanently remove the UK from the dataframe "Eudata"
94
95 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
96 ... January 2026.
97 #     Permanently update the dataframe "Eudata" accordingly.
98
99
100 #####
101 # 3: Simulation and probability (15 points) #
102 #####
103 # 3.1 Use R to produce one roll of a dice.
104 dice=sample(1:6,1)
105 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
106 k <- sample(1:6,1000,replace = TRUE)
107 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
108 #     a "4" or "5" from a dice
109
110 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
111 ... a dice
112 #     Using "k" and "m" from (3.2) and (3.4), estimate the expected value
113 #     and variance of the random variable z = 2k-m
114
115 # 3.5 Assume the yearly stock return to be normally distributed with a
116 ... mean of 0.12 and
117 #     a standard deviation of 0.2. Create a variable "stock" with 100
118 ... draws of stock returns
119
120 # 3.6 What is the probability of a negative stock return?
121 #     Answer this question ...
122
123 #####
124 # 4: Functions and Optimization (25 points)#
125 #####
126 # 4.1 Create the function f(x)=x^2 in R
127 f <- function(x){x^2}
128 # 4.2 Calculate the value of f for x=1
129 f(1)
130 # 4.3 Create a plot of the function for the interval [-2, 2]
131 #     If in doubt, type "?plot" to get the help file for the function
132
133 # 4.4 Numerically, by using R, find the location of the minimum using the
134 ... optim
135 #     function. Store the result of your minimization (the location of
136 ... the minimum)
137 #     in a variable called xmin
```

```
136
137 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
138 #       yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
139 #       Store the result of your minimization (the location of the minimum)
... in a
140 #       variable called xmin_grid
141
142 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
143 #       are not identical. Why?
144
145
146 y#####
147 # 5: Functions and algorithms (25 points) #
148 #####
149 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
150 #       like this
151 #       a) process individual digits from right to left
152 #       b) leave digits number 1,3,5 etc (counted from right) unchanged
153 #       c) multiply digits 2,3,6 etc (counted from right) by 2
154 #       d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
155 #       e) calculate the sum of all (processed) digits
156 #           IF the result can be divided by 10, the number is a valid credit
... card number
157 #
158 #       Example: 63487
159 #           Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
160 #                           multiply 3 by 2 and do not subtract anything, do
... not change 6
161 #                           7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
162 #       Hint: The operation x %% y yields the remainder of the division
... x/y.
163 #           For instance, 7 %% 4 gives 3
164
165 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
166 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
167
168 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
169 # You can use this to test your function
170 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
171 # right to left.
172 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
```

173
174
175

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # Spiatta Damiano
31
32
33 ######
34 # 1: R basics      (13 points)          #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1<- 10:1
39 v1
40 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
41 ... not use the c() command.
42 v2 <- seq(3,30, by = 3)
43 v2
44 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
45 M <- matrix(1:300, nrow = 10, ncol = 30)
46 head(M)
47 # 1.4 Print the first row of M
48 print(M[1,])
49 # 1.5 Display the last element of v1. Tell R to "display the last element
```

```
47... of x",
48 #      regardless of the dimension of v1.
49 print(v2[length(v2)])
50 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
51 (12/(19-7)^(1/5)
52 (log(1)+log(2))/((pi+1)/(pi-1))
53 log(sin(2)/exp(2))
54 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
55 u <- 10^(-5)
56 # 1.8 If you would now run the line
57 # print(U)
58 # you would get an error message. Which important concept of R is the
... reason
59 # for this error? Answer in 1 sentence as a comment.
60
61 #R in case sensitive, capital letters are not equals as normal letter. v1
... is different than V1
62
63 #####
64 # 2: Data and Logical conditions (20 points) #
65 #####
66 # The file "Eudata.csv" contains data about the (still 28) EU countries.
67 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
68 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
69
70 # The following line loads the data into an R dataframe
71 # Hint: Use Session/Set Working Directory/To Source File Location
72 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
73
74 # 2.1 How many countries are there in the dataset?
75 sum()
76 # 2.2 Calculate the total population of the EU
77 sum(myData$Population)
78 # 2.3 Print the population of the smallest and largest EU country by Area
79 myData$Population[which.min(myData$Area)]
80 myData$Population[which.max(myData$Area)]
81 # 2.4 Calculate the number of countries that are members of the Eurozone
82 sum(myData$IsEurozone == 1)
83 # 2.5 Calculate the total GDP of all Eurozone members
84 sum(myData$GDP[myData$IsEurozone == 1])
85 # 2.6 Calculate the GDP per capita in euros
86 #(a) of the total EU,
87 sum(myData$GDP) / sum(myData$Population)
88 #(b) of the Eurozone
89 sum(myData$GDP[myData$IsEurozone == 1]) /
...     sum(myData$Population[myData$IsEurozone == 1])
90 #(c) of the non-Eurozone EU
91 sum(myData$GDP[myData$IsEurozone != 1]) /
...     sum(myData$Population[myData$IsEurozone != 1])
92 # 2.7 When was the EU founded?
```

```
95 #      Hint: this must be the earliest year in which any country became a
96 ... member
97 myData$Accession == min(myData$Accession[myData$IsEurozone != 1])
98 # 2.8 Calculate the number of EU founding members
99 EU_founders <- sum(myData$Accession == 1953)
100 # 2.9 Only now you discover that the data set still contains the UK.
101 #      Permanently remove the UK from the dataframe "Eudata"
102 Eudata[Eudata$CountryName != "United Kingdom", ]
103 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
104 ... January 2026.
105 #      Permanently update the dataframe "Eudata" accordingly.
106 Eudata$Eurozone[Eudata$CountryName == "Bulgaria"] <- TRUE
107 #####
108 # 3: Simulation and probability (15 points) #
109 #####
110 # 3.1 Use R to produce one roll of a dice.
111 dice <- sample(1:6, size = 1, replace = TRUE)
112 dice
113 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
114 n <- 1000
115 k <- c(sample(1:6, size = n, replace = TRUE))
116 k
117 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
118 #      a "4" or "5" from a dice
119 prob_4_or_5 <- mean(k %in% c(4, 5))
120 prob_4_or_5
121 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
122 ... a dice
123 #      Using "k" and "m" from (3.2) and (3.4), estimate the expected value
124 #      and variance of the random variable z = 2k-m
125 m <- sample(1:6, size = n, replace = TRUE)
126 z <- 2*k - m
127
128 expected_value_z <- mean(z)
129 expected_value_z
130 variance_z <- var(z)
131 variance_z
132 # 3.5 Assume the yearly stock return to be normally distributed with a
133 ... mean of 0.12 and
134 #      a standard deviation of 0.2. Create a variable "stock" with 100
135 ... draws of stock returns
136 stock <- rnorm(100, mean = 0.12, sd = 0.2)
137 # 3.6 What is the probability of a negative stock return?
138 #      Answer this question ...
139 #      (a) by using the variable "stock" from (3.5)
140 prob_neg_emp <- mean(stock<0)
141 print(prob_neg_emp)
142 #      (b) by calculating the (theoretical) probability for a normal
143 ... distribution
144 prob_neg_theor <- pnorm(0,mean = 0.12, sd = 0.2)
```

```
141 print(prob_neg_theor)
142
143 ##### # 4: Functions and Optimization      (25 points)#
144 ##### # 4.1 Create the function f(x)=x^2 in R
145
146 f <- function(x){
147   x^2
148 }
149
150 # 4.2 Calculate the value of f for x=1
151 f(1)
152 # 4.3 Create a plot of the function for the interval [-2, 2]
153 #   If in doubt, type "?plot" to get the help file for the function
154 x <- seq(-2, 2, length.out = 200)
155 y <- f(x)
156 plot(x, y, type = "l", lwd = 2, main = "Plot of f(x) = x^2", xlab = "x",
157 ... ylab = "f(x)")
158 # 4.4 Numerically, by using R, find the location of the minimum using the
159 ... optim
160 #   function. Store the result of your minimization (the location of
161 ... the minimum)
162 #   in a variable called xmin
163 opt <- optim(0,f,method = "BFGS")
164 x_min = opt$par
165 y_min = opt$value
166 xmin <- list("min x" = x_min, "min y" = y_min)
167 print(xmin)
168
169 # 4.5 Now try to find the location of the minimum by implementing a grid
170 ... search
171 #   yourself. Choose N=100 grid points. Search in an interval between
172 ... -2 and 2.
173 #   Store the result of your minimization (the location of the minimum)
174 ... in a
175 #   variable called xmin_grid
176 N <- 100
177 grid <- seq(-2, 2, length.out = N)
178 values <- f(grid)
179
180 x_min_grid <- grid[which.min(values)]
181 y_min_grid <- min(values)
182
183 xmin_grid <- list("min x" = x_min_grid, "min y" = y_min_grid)
184 print(xmin_grid)
185
186
187 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
188 ... and (4.5)
189 #   are not identical. Why?
190
191
192 # The results are not identical because optim searches in a continuous
193 ... space,
194 # while the grid search is limited to a finite set of discrete points.
```

```
185
186 #####
187 # 5: Functions and algorithms      (25 points)      #
188 #####
189 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
190 #     like this
191 #     a) process individual digits from right to left
192 #         b) leave digits number 1,3,5 etc (counted from right) unchanged
193 #         c) multiply digits 2,3,6 etc (counted from right) by 2
194 #         d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
195 #             e) calculate the sum of all (processed) digits
196 #                 IF the result can be divided by 10, the number is a valid credit
... card number
197 #
198 #                 Example: 63487
199 #                     Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
200 #                         multiply 3 by 2 and do not subtract anything, do
... not change 6
201 #                         7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
202 #                 Hint: The operation x % y yields the remainder of the division
... x/y.
203 #                     For instance, 7 % 4 gives 3
204
205 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
206 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
207
208 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
209 # You can use this to test your function
210 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
211 # right to left.
212 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
213
214 checkLuhn <- function(digits) {
215   rev_digits <- rev(digits)
216
217   for (i in seq_along(rev_digits)) {
218     if (i %% 2 == 0) {
219       rev_digits[i] <- rev_digits[i] * 2
220       if (rev_digits[i] > 9) {
221         rev_digits[i] <- rev_digits[i] - 9
222       }
223     }
224   }
225 }
```

```
226     sum(rev_digits) %% 10 == 0
227 }
228
229 #test
230 checkLuhn(10)
231 checkLuhn(36)
232
233
234
235
236
237
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 #####0#####
27 # 0: Your name      (2 points)          #
28 #####0#####
29 # 0.1 Write your name below as a comment
30 # <your name>
31 #Eleonora Moroni
32
33 #####1#####
34 # 1: R basics      (13 points)         #
35 #####1#####
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 <- seq(10,1, -1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 <- seq(3,30,3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 row1 <- seq(1,291,10)
44 row2 <- seq(2,292,10)
45 row3 <- seq(3,293,10)
46 row4 <- seq(4,294,10)
47 row5 <- seq(10,300,10)
48 M <- rbind(row1,row2,row3,row4,row5)
49 # 1.4 Print the first row of M
```

```
48 M["row1",]  
49 # 1.5 Display the last element of v1. Tell R to "display the last element  
... of x",  
50 #      regardless of the dimension of v1.  
51 x<-v1  
52 tail(x,1)  
53 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.  
54 (12/(19-7))^^(1/5)  
55 (log(1)+log(2))/((pi+1)/(pi-1))  
56 log(sin(2)/exp(2))  
57 # 1.7 Create a variable "u" with the value "ten to the power minus 5"  
58 u<-(10)^(-5)  
59 # 1.8 If you would now run the line  
60 # print(U)  
61 # you would get an error message. Which important concept of R is the  
... reason  
62 # for this error? Answer in 1 sentence as a comment.  
63 #because u and U are seen in R as two diffrent variables, R is very  
... sensitivie  
64  
65 #####  
66 # 2: Data and Logical conditions (20 points) #  
67 #####  
68 # The file "Eudata.csv" contains data about the (still 28) EU countries.  
69 # The columns are: County Name, Code, Capital, Accession (=Year of  
... membership),  
70 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if  
... member)  
71  
72 # The following line loads the data into an R dataframe  
73 # Hint: Use Session/Set Working Directory/To Source File Location  
74 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)  
75  
76 # 2.1 How many countries are there in the dataset?  
77 #28  
78 # 2.2 Calculate the total population of the EU  
79 sum(Eudata$Population)  
80 # 2.3 Print the population of the smallest and largest EU country by Area  
81 Eudata$Population[min(Eudata$Area)]  
82 Eudata$Population[max(Eudata$Area)]  
83  
84 # 2.4 Calculate the number of countries that are members of the Eurozone  
85 sum(Eudata$Eurozone==1)  
86 # 2.5 Calculate the total GDP of all Eurozone members  
87 sum(Eudata$GDP[Eudata$Eurozone==1])  
88 # 2.6 Calculate the GDP per capita in euros  
89 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone  
...  
EU  
...  
#a)  
90 sum(Eudata$GDP)/sum(Eudata$Population)  
#b)  
91 sum(Eudata$GDP[Eudata$IsEurozone==1])/sum(Eudata$Population[Eudata$
```

```
93... IsEurozone==1])  
94 # 2.7 When was the EU founded?  
95 # Hint: this must be the earliest year in which any country became a  
... member  
96 min(Eudata$Accession)  
97  
98 # 2.8 Calculate the number of EU founding members  
99 sum(Eudata$Accession==1953)  
100 # 2.9 Only now you discover that the data set still contains the UK.  
101 # Permanently remove the UK from the dataframe "Eudata"  
102 Eudata<-Eudata[Eudata$Country != "UK"]  
103 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st  
... January 2026.  
104 # Permanently update the dataframe "Eudata" accordingly.  
105  
106  
107  
108 #####  
109 # 3: Simulation and probability (15 points) #  
110 #####  
111 # 3.1 Use R to produce one roll of a dice.  
112 sample(1:6,1)  
113 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice  
114 k<-sample(1:6,1000, replace=TRUE)  
115 # 3.3 Using "k" from (3.2), estimate the probability of obtaining  
116 # a "4" or "5" from a dice  
117 prob_4_or_5<- function(k)  
118 {sum(k%in% c(4,5))/  
119 length(k)}  
120 prob_4_or_5(k)  
121 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of  
... a dice  
122 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value  
123 # and variance of the random variable z = 2k-m  
124 m<-sample(1:6,1000,replace=TRUE)  
125 z<-2*k-m  
126 # 3.5 Assume the yearly stock return to be normally distributed with a  
... mean of 0.12 and  
127 # a standard deviation of 0.2. Create a variable "stock" with 100  
... draws of stock returns  
128 stock<-rnorm(100,0.12,0.2)  
129 # 3.6 What is the probability of a negative stock return?  
130 # Answer this question ...  
131 # (a) by using the variable "stock" from (3.5)  
132 # (b) by calculating the (theoretical) probability for a normal  
... distribution  
133 negative_stock<-sum(stock<0)/length(stock)  
134 print(negative_stock)  
#####  
135 # 4: Functions and Optimization (25 points)  
#####  
136 # 4.1 Create the function f(x)=x^2 in R
```

```
139 f<-function(x) {  
140   return<-x^2  
141 }  
142 # 4.2 Calculate the value of f for x=1  
143 f(1)  
144 # 4.3 Create a plot of the function for the interval [-2, 2]  
145 #     If in doubt, type "?plot" to get the help file for the function  
146 x<-seq(-2,2)  
147 plot(x,f(x), "l")  
148 # 4.4 Numerically, by using R, find the location of the minimum using the  
149 ... optim  
150 #     function. Store the result of your minimization (the location of  
151 ... the minimum)  
152 #     in a variable called xmin  
153 xmin<-function(x){  
154   stopifnot(-2<=x && x<=2)  
155   return(x)  
156 }  
157 # 4.5 Now try to find the location of the minimum by implementing a grid  
158 ... search  
159 #     yourself. Choose N=100 grid points. Search in an interval between  
160 ... -2 and 2.  
161 #     Store the result of your minimization (the location of the minimum)  
162 ... in a  
163 #     variable called xmin_grid  
164  
165 #####  
166 # 5: Functions and algorithms      (25 points)      #  
167 #####  
168 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
169 ... is valid. It goes  
170 #     like this  
171 #     a) process individual digits from right to left  
172 #     b) leave digits number 1,3,5 etc (counted from right) unchanged  
173 #     c) multiply digits 2,3,6 etc (counted from right) by 2  
174 #     d) if a digit (after multiplying by 2) is larger than 9, subtract  
175 #         9  
176 #     e) calculate the sum of all (processed) digits  
177 #     IF the result can be divided by 10, the number is a valid credit  
178 ... card number  
179 #  
177 #     Example: 63487  
178 #     Right to left: do not change 7, multiply 8 by 2 and subtract 9, do  
179 ... not change 4,  
#                         multiply 3 by 2 and do not subtract anything, do  
... not change 6
```

```
180 # 7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
181 # Hint: The operation x %% y yields the remainder of the division
... x/y.
182 # For instance, 7 %% 4 gives 3
183
184 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
185 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
186
187 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
188 # You can use this to test your function
189 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
190 # right to left.
191 digits <- as.numeric(unlist(strsplit("5019717010103742", "")))
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # Erica Trofimov
31
32
33 ######
34 # 1: R basics      (13 points)          #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 <- seq(10, 1, -1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 <- seq(3, 30, 3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 v1 = seq(1, 291, 10)
44 v2 = seq(2, 292, 10)
45 v3 = seq(3, 293, 10)
46 v4 = seq(4, 294, 10)
47 v5 = seq(10, 300, 10)
48
49 M = rbind(v1, v2, v3, v4, v5)
```

```
48 # 1.4 Print the first row of M
49 M[1, ]
50 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
51 #      regardless of the dimension of v1.
52 v1[length(v1)]
53 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
54 ## 1.
55 ((12)/(19 -7 ))^(1 / 5)
56 ## 2.
57 (log10(1) + log10(2))/ ((pi + 1)/ (pi - 1))
58
59 ## 3.
60 log10 ( sin(2)/exp(2))
61
62 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
63
64 u <- 10^(-5)
65 # 1.8 If you would now run the line
66 # print(U)
67 # you would get an error message. Which important concept of R is the
... reason
68 # for this error? Answer in 1 sentence as a comment.
69 ## R is case-sensitive. Hence, it would consider U as a totally different
... object than U. We do not have any U object
70 ## in our directory, hence R will give error. If we want to print the
... value of the variable we just created,
71 ## we would need to run print(u).
72
73 ######
74 # 2: Data and Logical conditions (20 points) #
75 #####
76 # The file "Eudata.csv" contains data about the (still 28) EU countries.
77 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
78 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
79
80 # The following line loads the data into an R dataframe
81 # Hunt: Use Session/Set Working Directory/To Source File Location
82
83 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
84
85 # 2.1 How many countries are there in the dataset?
86 nrow(Eudata)
87 # 2.2 Calculate the total population of the EU
88 sum(Eudata$Population)
89 # 2.3 Print the population of the smallest and largest EU country by Area
90 min_pop = which.min(Eudata$Area)
91 max_pop = which.max(Eudata$Area)
92 Eudata$Population[min_pop]
93 Eudata$Population[max_pop]
```

```
94
95 # 2.4 Calculate the number of countries that are members of the Eurozone
96 sum(Eudata$Eurozone)
97 # 2.5 Calculate the total GDP of all Eurozone members
98 library(dplyr)
99 Eurozone%>%
100   filter ( Eurozone == 1) %>%
101   summarize( sum(GDP))
102 ## or
103 Eurozone = Eudata[ Eudata$Eurozone == 1, ]
104 sum(Eurozone$GDP)
105
106 # 2.6 Calculate the GDP per capita in euros
107 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
108 ...
109 EU
110 a = sum(Eudata$GDP)/sum(Eudata$Population)
111 b = sum(Eudata$GDP)/sum(Eurozone$Population)
112 NonEuro = Eudata[ Eudata$Eurozone == 0, ]
113 c = sum(Eudata$GDP)/sum(NonEuro$Population)
114
115
116
117 found_year = Eudata$Accession[which.min(Eudata$Accession) ]
118
119 # 2.8 Calculate the number of EU founding members
120 nrow( Eudata[ Eudata$Accession == 1953, ] )
121 # 2.9 Only now you discover that the data set still contains the UK.
122 #      Permanently remove the UK from the dataframe "Eudata"
123 Eudata[ Eudata$Code == 'UK', ]
124 Eudata[ - 28, ]
125
126 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
127 ...
128 January 2026.
129 #      Permanently update the dataframe "Eudata" accordingly.
130 Eudata[ Eudata$CountyName == 'Bulgaria', 'Eurozone'] <- 1
131 #####
132 # 3: Simulation and probability (15 points) #
133 #####
134 # 3.1 Use R to produce one roll of a dice.
135 sample(1 : 6, 1, replace = TRUE)
136 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
137 k <- sample(1:6, 1000, replace = TRUE)
138 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
139 #      a "4" or "5" from a dice
140 k4 = sum(k == 4)
141 k5 = sum(k == 5)
142 prob_k4_5 = (k4 + k5 ) / (length(k))
```

```
143
144 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
... a dice
145 #      Using "k" and "m" from (3.2) and (3.4), estimate the expected value
146 #      and variance of the random variable z = 2k-m
147 m = sample(1:6, 1000, replace = TRUE)
148
149 expr <- function(k,m){
150   return(2*k - m )
151 }
152 z = expr ( k, m)
153 z_mean = mean(z)
154 z_var = var(z)
155
156 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
157 #      a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
158 stock <- rnorm(100, 0.12, 0.3)
159
160
161 # 3.6 What is the probability of a negative stock return?
162 #      Answer this question ...
163 #      (a) by using the variable "stock" from (3.5)
164 neg_stock = sum( stock < 0)
165 prob_neg = neg_stock / length(stock)
166 prob_neg
167 #      (b) by calculating the (theoretical) probability for a normal
... distribution
168 pnorm( 0 , neg_stock , lower.tail = TRUE)
169
170 ######
171 # 4: Functions and Optimization      (25 points)#
172 #####
173 # 4.1 Create the function f(x)=x^2 in R
174 f = function(x){
175   return ( x^2)
176 }
177 # 4.2 Calculate the value of f for x=1
178 f(1)
179 # 4.3 Create a plot of the function for the interval [-2, 2]
180 plot( f, xlim = c(-2,2))
181 #      If in doubt, type "?plot" to get the help file for the function
182
183 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
184 #      function. Store the result of your minimization (the location of
... the minimum)
185 #      in a variable called xmin
186 opt = optim(
187   par = 2,
188   f,
```

```
189     method = 'BFGS'  
190     )  
191 opt$par  
192 xmin = opt$value  
193 f(xmin)  
194 # 4.5 Now try to find the location of the minimum by implementing a grid  
... search  
195 #       yourself. Choose N=100 grid points. Search in an interval between  
... -2 and 2.  
196 #       Store the result of your minimization (the location of the minimum)  
... in a  
197 #       variable called xmin_grid  
198 N = 100  
199 points = seq( -2, 2 , 0.04)  
200 y_grid = sapply( points, f)  
201 min_index = which.min(y_grid)  
202 best_y_grid_search = y_grid[ min_index]  
203 xmin_grid = x_grid [ min_index]  
204  
205 min_coordinates = c(xmin_grid, best_y_grid_search )  
206 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)  
... and (4.5)  
207 #       are not identical. Why?  
208 ## Results of 4.4 depend on the arbitrary parameter we want the optim()  
... function  
209 ## to start iterating from. Based on that, results slightly vary.  
... Moreover, it also depends on the algorithm we chose.  
210 ## Since the result of this minimization problem is 0, choosing a Gradient  
... descent or a Nelder-Mead could  
211 ## give slightly different results (but still right if we approximate)  
212  
213 ## The grid search, instead, is a brute force method. It is still  
... effective,  
214 ## but highly depends on the value we decide to look from. If 0 was not  
... in the grid, and we would have added, say, 0.1 as the value  
215 ## closer to 0, then this method would have given us 0.1 as xmin value,  
... completely ignoring 0 which was not in our grid.  
216  
217 y#####  
218 # 5: Functions and algorithms      (25 points)      #  
219 #####  
220 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
... is valid. It goes  
221 #       like this  
222 #       a) process individual digits from right to left  
223  
224 #       b) leave digits number 1,3,5 etc (counted from right) unchanged  
225  
226 #       c) multiply digits 2,3,6 etc (counted from right) by 2  
227  
228  
229 #       d) if a digit (after multiplying by 2) is larger than 9, subtract
```

```
229... 9
230 #      e) calculate the sum of all (processed) digits
231 #          IF the result can be divided by 10, the number is a valid credit
232 ... card number
233 #
234 #          Example: 63487
235 #              Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
236 ... not change 4,
237 #                  multiply 3 by 2 and do not subtract anything, do
238 ... not change 6
239 #                  7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
240 ... valid
241 #          Hint: The operation x %% y yields the remainder of the division
242 ... x/y.
243 #          For instance, 7 %% 4 gives 3
244
245 checkLuhn <- function(digits) {
246   right_digits <- rev(digits)
247
248   for (i in 1:length(right_digits)) {
249     if (i %% 2 == 0) {
250       doubled <- right_digits[i] * 2
251       if (doubled > 9) {
252         doubled = doubled - 9
253       }
254       right_digits[i] <- doubled
255     }
256   }
257   sum <- sum(right_digits)
258
259   if (sum %% 10 == 0) {
260     return(TRUE)
261   } else {
262     return(FALSE)
263   }
264   result <- checkLuhn(card_to_check)
265 }
266
267 # Write a function called checkLuhn that takes as argument a vector of
268 ... individual digits
269 # and returns TRUE if the number is a valid number and FALSE if it is not
270 ... valid.
271
272
273
274
275 # The following line takes a *valid* credit card number and creates a
276 ... vector with single digits.
277 # You can use this to test your function
278 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
279 ... algorithm works from
280 # right to left.
281 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
282 checkLuhn(digits )
283
284
```

272
273

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Filippo Manachino (2 points) #
28 #####
29 # 0.1 Write your name below as a comment
30 # <Filippo Manachino>
31
32
33 #####
34 # 1: R basics (13 points) #
35 #####
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 <- 10:1
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 <- seq(3, 30, by=3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 M <- matrix(1:300, nrow=10, ncol=30)
44 # 1.4 Print the first row of M
45 M[1, ]
46 # 1.5 Display the last element of v1. Tell R to "display the last element
47 ... of x",
48 #     regardless of the dimension of v1.
49 v1[length(v1)]
```

```
47 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.  
48 ((sqrt^5(12/(19-17))))  
49 (log(1)+log(2))/((pi+1)/(pi-1))  
50 log((sin(2))/(exp(1))^2)  
51 # 1.7 Create a variable "u" with the value "ten to the power minus 5"  
52 u <- 10^(-5)  
53 # 1.8 If you would now run the line  
54 # print(U)  
55 # you would get an error message. Which important concept of R is the  
... reason  
56 # for this error? Answer in 1 sentence as a comment.  
57 #R is case-sensitive: U and u are different objects (different names)  
58  
59 #####  
60 # 2: Data and Logical conditions (20 points) #  
61 #####  
62 # The file "Eudata.csv" contains data about the (still 28) EU countries.  
63 # The columns are: County Name, Code, Capital, Accession (=Year of  
... membership),  
64 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if  
... member)  
65  
66 # The following line loads the data into an R dataframe  
67 # Hint: Use Session/Set Working Directory/To Source File Location  
68 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)  
69  
70 # 2.1 How many countries are there in the dataset?  
71 nrow(Eudata) #this is just the command, to save this into a variable just  
... do n_countries <- nrow(Eudata)  
72 # 2.2 Calculate the total population of the EU  
73 total_pop <- sum(Eudata$Population)  
74 # 2.3 Print the population of the smallest and largest EU country by Area  
75 pop_smallest_by_area <- Eudata$Population[which.min(Eudata$Area)]  
76 pop_biggest_by_area <- Eudata$Population[which.max(Eudata$Area)]  
77 # 2.4 Calculate the number of countries that are members of the Eurozone  
78 n_eurozone <- sum(Eudata$Eurozone==1)  
79 # 2.5 Calculate the total GDP of all Eurozone members  
80 gdp_eurozone <- sum(Eudata$GDP[Eudata$Eurozone==1])  
81 # 2.6 Calculate the GDP per capita in euros  
82 # (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone  
EU  
...  
83 total_gdp <- sum(Eudata$GDP)  
84 gdp_eurozone <- sum(Eudata$GDP[Eudata$Eurozone==1])  
85 gdp_non_eurozone <- sum(Eudata$GDP[Eudata$Eurozone==0])  
86  
87 total_gdp_pc <- total_gdp/sum(Eudata$Population)  
88 gdp_pc_eu <- gdp_eurozone/sum(Eudata$Population[Eudata$Eurozone==1])  
89 gdp_pc_non_eu <-  
... gdp_non_eurozone/sum(Eudata$Population[Eudata$Eurozone==0])  
90 # 2.7 When was the EU founded?  
91 # Hint: this must be the earliest year in which any country became a  
member
```

```
92 eu_founding <- min(Eudata$Accession)
93 # 2.8 Calculate the number of EU founding members
94 total_founders <- sum(Eudata$Accession==1953)
95 # 2.9 Only now you discover that the data set still contains the UK.
96 # Permanently remove the UK from the dataframe "Eudata"
97 Eudata <- subset(Eudata, Code != "UK")
98 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
... January 2026.
99 # Permanently update the dataframe "Eudata" accordingly.
100
101
102 ######
103 # 3: Simulation and probability (15 points) #
104 #####
105 # 3.1 Use R to produce one roll of a dice.
106 dice <- sample(1:6, 1)
107 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
108 k <- sample(1:6, 1000, replace = TRUE)
109 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
110 # a "4" or "5" from a dice
111 p4_p5 <- mean(k==4 | k==5)
112 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
... a dice
113 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
114 # and variance of the random variable z = 2k-m
115 m <- sample(1:6, 1000, replace = TRUE)
116 z <- 2*k-m
117 Expected <- mean(z)
118 Variance <- var(z)
119 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
120 # a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
121 stock <- rnorm(n = 100, mean = 0.12, sd = 0.2)
122 # 3.6 What is the probability of a negative stock return?
123 # Answer this question ...
124 # (a) by using the variable "stock" from (3.5)
125 p_neg_hat <- mean(stock<0)
126 # (b) by calculating the (theoretical) probability for a normal
... distribution
127 p_neg_theory <- pnorm(0, mean = 0.12, sd = 0.2)
128
129 #####
130 # 4: Functions and Optimization (25 points)#
131 #####
132 # 4.1 Create the function f(x)=x^2 in R
133 f <- function(x){x^2}
134 # 4.2 Calculate the value of f for x=1
135 f(1)
136 # 4.3 Create a plot of the function for the interval [-2, 2]
137 # If in doubt, type "?plot" to get the help file for the function
138 x <- seq(-2, 2, length.out=100)
```

```
139 plot(x,f(x),type="l")
140 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
141 #     function. Store the result of your minimization (the location of
... the minimum)
142 #     in a variable called xmin
143 xmin <- optim(0, f, method = "L-BFGS-B",lower=-2, upper=2)
144 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
145 #     yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
146 #     Store the result of your minimization (the location of the minimum)
... in a
147 #     variable called xmin_grid
148 x <- seq(-2,2,length.out=100)
149 xmin_grid <- grid()
150 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
151 #     are not identical. Why?
152
153
154 y#####
155 # 5: Functions and algorithms (25 points) #
156 #####
157 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
158 #     like this
159 #     a) process individual digits from right to left
160 #         b) leave digits number 1,3,5 etc (counted from right) unchanged
161 #         c) multiply digits 2,3,6 etc (counted from right) by 2, the
... sequence in ambiguous, i used 2,4,6 etc
162 #         d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
163 #         e) calculate the sum of all (processed) digits
164 #             IF the result can be divided by 10, the number is a valid credit
... card number
165 #
166 #     Example: 63487
167 #             Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
168 #                         multiply 3 by 2 and do not subtract anything, do
... not change 6
169 #                         7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
170 #             Hint: The operation x %% y yields the remainder of the division
... x/y.
171 #                 For instance, 7 %% 4 gives 3
172
173 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
174 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
```

```
175
176 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
177 # You can use this to test your function
178 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
179 # right to left.
180 #the sequence is ambiguous, i used 2,4,6 etc
181 digits <- as.numeric(unlist(strsplit("5019717010103742", "")))
182 d <- c(2,4,7,3,0,1,0,1,0,7,1,7,9,1,0,5)
183 checkLuhn <- function(d){digits}
184 for (i in 1:16){
185   if (d=="1,3,5,7,9"){
186     creditnumber <- d
187   }
188   else if(d=="2,4,6,8"){
189     h <- d*2
190     if(h>9){
191       h <- h-9
192       creditnumber <- h
193     }
194   }
195 }
196 #unfortunately i don't remember how to code it, the base idea of my
... algorithm was to create a for cycle that would scan the vector number by
... number using an if condition
197 #if the number was 1,3,5,7,9 it would add it to a vector called
... creditnumber, else if the number was 2,4,6,8 it would first multiply it
... by 2 then with the other if it would check
198 #if the number was >9 in that case it would divide by 2 and then add it
... to the vector
199 #at the end outside the vector i would have summed it and checked if the
... division by 10 would leave an integer or no so to determine if the credit
... number was valid or no
200
201
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ##### #0: Your name (2 points) #
27 ##### #0.1 Write your name below as a comment
28 ##### #<your name>
29
30 ###### <Leonardo Zanga>
31 ###### #1: R basics (13 points) #
32
33 ##### #1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
34 ... not use the c() command.
35 v1<-seq(10,1,-1)
36 v1
37
38 ##### #1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
39 ... not use the c() command.
40 v2<-3*seq(1:10)
41 v2
42 ##### #1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 M<-rbind(seq(1,291,10),seq(2,292,10),seq(3,293,10),seq(4,294,10),seq(10,
44 ... 300,10))
45 M
46 # 1.4 Print the first row of M
47 M[1,]
```

```
47 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
48 #      regardless of the dimension of v1.
49 M[length(M)]
50 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
51 (12/(19-7))^1/5
52 (log(1)+log(2))/((pi+1)/(pi+2))
53 log(sin(2)/exp(2))
54 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
55 u<-10^(-5)
56 # 1.8 If you would now run the line
57 # print(U)
58 # you would get an error message. Which important concept of R is the
... reason
59 # for this error? Answer in 1 sentence as a comment.
60 ##### Because U and u are different values in R
61
62
63 #####
64 # 2: Data and Logical conditions (20 points) #
65 #####
66 # The file "Eudata.csv" contains data about the (still 28) EU countries.
67 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
68 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
69
70 # The following line loads the data into an R dataframe
71 # Hint: Use Session/Set Working Directory/To Source File Location
72 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
73
74 # 2.1 How many countries are there in the dataset?
75 nrow(Eudata)
76
77 # 2.2 Calculate the total population of the EU
78 sum(Eudata$Population[Eudata$Eurozone==1])
79 # 2.3 Print the population of the smallest and largest EU country by Area
80 Eudata$Population[which.max(Eudata$Area)]
81 Eudata$Population[which.min(Eudata$Area)]
82 # 2.4 Calculate the number of countries that are members of the Eurozone
83 sum(Eudata$Eurozone==TRUE)
84 # 2.5 Calculate the total GDP of all Eurozone members
85 sum(Eudata$GDP[Eudata$Eurozone])
86
87 # 2.6 Calculate the GDP per capita in euros
88 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
... EU
89 sum(Eudata$GDP/Eudata$Population)
90 sum(Eudata$GDP[Eudata$Eurozone==TRUE]/Eudata$Population[Eudata$Eurozone==
... TRUE])
91 sum(Eudata$GDP[Eudata$Eurozone==FALSE]/Eudata$Population[Eudata$Eurozone=
... =FALSE])
```

```
92 # 2.7 When was the EU founded?  
93 # Hint: this must be the earliest year in which any country became a  
... member  
94 min(Eudata$Accesion)  
95  
96 # 2.8 Calculate the number of EU founding members  
97 sum(Eudata$CountyName[Eudata$Accesion==1953])  
98 # 2.9 Only now you discover that the data set still contains the UK.  
99 # Permanently remove the UK from the dataframe "Eudata"  
100 Eudata<-Eudata[Eudata$countyname=='United Kingdom',]  
101 Eudata  
102  
103 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st  
... January 2026.  
104 # Permanently update the dataframe "Eudata" accordingly.  
105 new<-bulgaria  
106 Eudata$New  
107  
108  
109 #####  
110 # 3: Simulation and probability (15 points) #  
111 #####  
112 # 3.1 Use R to produce one roll of a dice.  
113 R<-sample(1:6,1,replace=TRUE)  
114 R  
115 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice  
116 k<-sample(1:6,100,replace=TRUE)  
117 k  
118 # 3.3 Using "k" from (3.2), estimate the probability of obtaining  
119 # a "4" or "5" from a dice  
120 k[k>=4 & k<=5]  
121 length(k[k>=4 & k<=5])/length(k)  
122  
123 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of  
... a dice  
124 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value  
125 # and variance of the random variable z = 2k-m  
126 m<-sample(1:6,1000,replace=TRUE)  
127 m  
128 z=2*k-m  
129 mean(z)  
130  
131 # 3.5 Assume the yearly stock return to be normally distributed with a  
... mean of 0.12 and  
132 # a standard deviation of 0.2. Create a variable "stock" with 100  
... draws of stock returns  
133 stocks<-rnorm(100,0.12,0.2)  
134 stocks  
135 # 3.6 What is the probability of a negative stock return?  
136 # Answer this question ...  
137 # (a) by using the variable "stock" from (3.5)  
138 prob<-mean(stocks<0)
```

```
139 prob
140 #      (b) by calculating the (theoretical) probability for a normal
... distribution
141 pnorm(0,0.12,0.2)
142
143 #####
144 # 4: Functions and Optimization      (25 points)#
145 #####
146 # 4.1 Create the function f(x)=x^2 in R
147 f<-function(x){
148   ris<-x^2
149   print(ris)
150 }
151 # 4.2 Calculate the value of f for x=1
152 f(1)
153
154 # 4.3 Create a plot of the function for the interval [-2, 2]
155 #      If in doubt, type "?plot" to get the help file for the function
156 plot(f,2,-2)
157 plot()
158 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
159 #      function. Store the result of your minimization (the location of
... the minimum)
160 #      in a variable called xmin
161 xmin<-optim( par=0, fn=f, method="BFGS")
162 xmin
163 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
164 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
165 #      Store the result of your minimization (the location of the minimum)
... in a
166 #      variable called xmin_grid
167 lowerbound=-2
168 upperbound=+2
169 xmin_grid<-optim( par=100, fn=f, method="BFGS",lower=lowerbound,
... upper=upperbound)
170 xmin
171 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
172 #      are not identical. Why?
173 because i have add upper and lower bound
174
175
176 y#####
177 # 5: Functions and algorithms      (25 points)      #
178 #####
179 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
180 #      like this
181 #      a) process individual digits from right to left
```

```
182 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
183 #      c) multiply digits 2,3,6 etc (counted from right) by 2
184 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
185 #          9
186 #      e) calculate the sum of all (processed) digits
187 #      IF the result can be divided by 10, the number is a valid credit
188 #      card number
189 #
190 #      Example: 63487
191 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
192 #          not change 4,
193 #                  multiply 3 by 2 and do not subtract anything, do
194 #          not change 6
195 #                  7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
196 #          valid
197 #      Hint: The operation x %% y yields the remainder of the division
198 #          x/y.
199 #                  For instance, 7 %% 4 gives 3
200
201 # Write a function called checkLuhn that takes as argument a vector of
202 # individual digits
203 # and returns TRUE if the number is a valid number and FALSE if it is not
204 # valid.
205 s=0
206 v1<-c(6,3,4,8,7)
207 n<-v1()
208
209 chwcluLuhn<-function(v1){
210   if(n*2>9){}
211   ris<-n+((n+1)*2-9)+4+3*2+6/10
212   valid<-(ris/10)==TRUE
213   s=s+s1
214 }
215 else{
216
217 }# The following line takes a *valid* credit card number and creates a
# vector with single digits.
# You can use this to test your function
# Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
# algorithm works from
# right to left.
218 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 #####0#####
27 # 0: Your name      (2 points)          #
28 #####0#####
29 # 0.1 Write your name below as a comment
30 # Dell Erba Lisa
31
32
33 #####1#####
34 # 1: R basics      (13 points)         #
35 #####1#####
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1<-10:1
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2<-seq(3,30,by=3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 row1<-seq(1,291,by=10)
44 row2<-seq(2,292,by=10)
45 row3<-seq(3,293,by=10)
46 row4<-seq(4,292,by=10)
47 row5<-seq(10,300,by=10)
48 M<-rbind(row1,row2,row3,row4,row5)
49 # 1.4 Print the first row of M
```

```
48 M[1,]
49 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
50 #      regardless of the dimension of v1.
51 tail(v1,1)
52 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
53 (12/(19-7))^^(1/5)
54 (log10(1)+log10(2))/((pi+1)/(pi-1))
55 log10(sin(2)/exp(2))

56
57 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
58 u<-10^(-5)
59 # 1.8 If you would now run the line
60 # print(U)
61 # you would get an error message. Which important concept of R is the
... reason
62 # for this error? Answer in 1 sentence as a comment.
63 #R is a case sensitive, our variable is called u and not U, which is
... undefined.

64
65 #####
66 # 2: Data and Logical conditions (20 points) #
67 #####
68 # The file "Eudata.csv" contains data about the (still 28) EU countries.
69 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
70 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)

71
72 # The following line loads the data into an R dataframe
73 # Hint: Use Session/Set Working Directory/To Source File Location
74 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
75 # 2.1 How many countries are there in the dataset?
76 nrow(Eudata)
77 # 2.2 Calculate the total population of the EU
78 tot_pop<-sum(Eudata$Population)
79 # 2.3 Print the population of the smallest and largest EU country by Area
80 min(Eudata$Population)
81 max(Eudata$Population)
82 # 2.4 Calculate the number of countries that are members of the Eurozone
83 sum(Eudata$Eurozone==1)
84 # 2.5 Calculate the total GDP of all Eurozone members
85 gdp_eurozone<-sum(Eudata$GDP[Eudata$Eurozone==1])
86 # 2.6 Calculate the GDP per capita in euros
87 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
... EU
88 gdp_tot<-sum(Eudata$GDP)
89 gdp_per_capita_eu<-gdp_tot/tot_pop
90 pop_eurozone<-sum(Eudata$Population[Eudata$Eurozone==1])
91 gdp_per_capita_eurozone<-gdp_eurozone/pop_eurozone
92 gdp_non_eurozone<-sum(Eudata$GDP[Eudata$Eurozone==0])
93 pop_non_eurozone<-sum(Eudata$Population[Eudata$Eurozone==0])
```

```
94 gdp_per_capita_non_eurozone<-gdp_non_eurozone/pop_non_eurozone
95 # 2.7 When was the EU founded?
96 # Hint: this must be the earliest year in which any country became a
97 ... member
98 min(Eudata$Accession)
99 # 2.8 Calculate the number of EU founding members
100 founding_memebers<-sum(Eudata$Accession==1953)
101 # 2.9 Only now you discover that the data set still contains the UK.
102 # Permanently remove the UK from the dataframe "Eudata"
103 Eudata<-Eudata[Eudata$CountyName!="United Kingdom", ]
104 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
105 ... January 2026.
106 # Permanently update the dataframe "Eudata" accordingly.
107 Eudata<-Eudata[Eudata$CountyName=="Bulgaria", ]
108 #####
109 # 3: Simulation and probability (15 points) #
110 #####
111 # 3.1 Use R to produce one roll of a dice.
112 dice<-sample(1:6,1,replace=TRUE)
113 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
114 k<-sample(1:6,1000,replace=TRUE)
115 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
116 # a "4" or "5" from a dice
117 mean(k==4 | k==5)
118 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
119 ... a dice
120 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
121 # and variance of the random variable z = 2k-m
122 m<-sample(1:6,1000,replace=TRUE)
123 z<-2*k-m
124 expected_value_z<-mean(z)
125 print(expected_value_z)
126 var(z)
127 # 3.5 Assume the yearly stock return to be normally distributed with a
128 ... mean of 0.12 and
129 # a standard deviation of 0.2. Create a variable "stock" with 100
130 ... draws of stock returns
131 stock<-rnorm(100,0.12,0.2)
132 # 3.6 What is the probability of a negative stock return?
133 # Answer this question ...
134 # (a) by using the variable "stock" from (3.5)
135 # (b) by calculating the (theoretical) probability for a normal
136 ... distribution
137
138 #a
139 mean(stock<0)
140 #b
141 stock<-pnorm(0,0.12,0.2)
142 #####
143 # 4: Functions and Optimization (25 points)#
144
```

```
140 #####  
141 # 4.1 Create the function f(x)=x^2 in R  
142 f<-function(x){  
143   x<-x^2  
144   return(x)  
145 }  
146 # 4.2 Calculate the value of f for x=1  
147 f(1)  
148 # 4.3 Create a plot of the function for the interval [-2, 2]  
149 #   If in doubt, type "?plot" to get the help file for the function  
150 curve(f, -2,2)  
151 # 4.4 Numerically, by using R, find the location of the minimum using the  
... optim  
152 #   function. Store the result of your minimization (the location of  
... the minimum)  
153 #   in a variable called xmin  
154 xmin<-optim(par=0,fn=f,lower=-2,upper=2,method='L-BFGS')  
155 # 4.5 Now try to find the location of the minimum by implementing a grid  
... search  
156 #   yourself. Choose N=100 grid points. Search in an interval between  
... -2 and 2.  
157 #   Store the result of your minimization (the location of the minimum)  
... in a  
158 #   variable called xmin_grid  
159 xmin<-optim(par=100,fn=f,method='Brent', lower=-2,upper=2)  
160  
161 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)  
... and (4.5)  
162 #   are not identical. Why?  
163 # Because of the floating-point approximation  
164 #The two ways to find the location of the minimum are close to each other  
... but they aren't exactly the same  
165 #####  
166 # 5: Functions and algorithms (25 points) #  
167 #####  
168 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
... is valid. It goes  
169 #   like this  
170 #   a) process individual digits from right to left  
171 #   b) leave digits number 1,3,5 etc (counted from right) unchanged  
172 #   c) multiply digits 2,3,6 etc (counted from right) by 2  
173 #   d) if a digit (after multiplying by 2) is larger than 9, subtract  
... 9  
174 #   e) calculate the sum of all (processed) digits  
175 #   IF the result can be divided by 10, the number is a valid credit  
... card number  
176 #  
177 #   Example: 63487  
178 #   Right to left: do not change 7, multiply 8 by 2 and subtract 9, do  
... not change 4,  
179 #   multiply 3 by 2 and do not subtract anything, do  
... not change 6
```

```
180 # 7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
181 # Hint: The operation x %% y yields the remainder of the division
... x/y.
182 # For instance, 7 %% 4 gives 3
183
184 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
185 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
186
187 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
188 # You can use this to test your function
189 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
190 # right to left.
191 digits <- as.numeric(unlist(strsplit("5019717010103742", "")))
192
193 checkLuhn<-function(CardNumber){
194   digits<-as.numeric(unlist(strsplit(CardNumber, "")))
195   digits_rev<-rev(digits)
196   for(i in seq_along(digits_rev)){
197     if( i %% 2==0){
198       digits_rev[i]<-digits_rev[i]*2
199       if(digits_rev[i]>9){
200         digits_rev[i]<-digits_rev[i]-9
201       }
202     }
203   }
204   total<-sum(digits_rev)
205   return(total%%10==0)
206 }
207
208
209 checkLuhn("5019717010103742")
210
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
... using R!
6 # - The exam takes 90 minutes and has 5 questions
7 # - There are a total of 100 points
8 # >>>> SUBMIT your answers on iCorsi in time
... <---
IMPORTANT!!!
9
10 # Material that you can use
11 # - the R help function
12 # - any printed material (open book)
13
14 # Notes
15 # - There are several ways how this exam can be solved.
16 # - What counts are ...
17 #   + that your program works
18 #   + that you follow the instructions
19 #   + that the program fulfills the requirements
20 # - We do not expect you to provide exactly the solution presented in
... class.
21
22 # Tip:
23 # - Use R to check your solution!
24
25
26 ######
27 # 0: Your name (2 points) #
28 #####
29 # 0.1 Write your name below as a comment
30 # <your name>
31 #Martinelli Ludovico
32
33 #####
34 # 1: R basics (13 points) #
35 #####
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
... not use the c() command.
37 v1<- seq(10,1,-1)
38 v1
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
... not use the c() command.
40 v2<-seq(3,30,3)
41 v2
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 riga1<- seq(1,291,10)
44 riga2<-seq(2,292,10)
45 riga3<-seq(3,293,10)
46 riga4<-seq(4,294,10)
47 riga10<-seq(10,300,10)
```

```
48 M<-matrix(c(riga1,riga2,riga3,riga4,riga10), nrow = 5, byrow = T)
49 M
50 # 1.4 Print the first row of M
51 M[1,]
52 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
53 #      regardless of the dimension of v1.
54 tail(v1,1)
55 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
56 (12/(19-7))^(1/5)
57 (log10(1)+log10(2))/((pi+1)/(pi-1))
58 log10(sin(2)/exp(2))
59 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
60 u<-10^12
61 # 1.8 If you would now run the line
62 # print(U)
63 # you would get an error message. Which important concept of R is the
... reason
64 # for this error? Answer in 1 sentence as a comment.
65
66 #R is case sensitive, so we have to pay attention when we write capital
... letter.
67
68 #####
69 # 2: Data and Logical conditions (20 points) #
70 #####
71 # The file "Eudata.csv" contains data about the (still 28) EU countries.
72 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
73 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
74
75 # The following line loads the data into an R dataframe
76 # Hint: Use Session/Set Working Directory/To Source File Location
77 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
78 View(Eudata)
79 # 2.1 How many countries are there in the dataset?
80 nrow(Eudata)
81 # 2.2 Calculate the total population of the EU
82 sum(Eudata$Population)
83 # 2.3 Print the population of the smallest and largest EU country by Area
84 Eudata$Population[which.min(Eudata$Area)]
85 Eudata$Population[which.max(Eudata$Area)]
86 # 2.4 Calculate the number of countries that are members of the Eurozone
87 sum(Eudata$Eurozone==1)
88 # 2.5 Calculate the total GDP of all Eurozone members
89 sum(Eudata$GDP[Eudata$Eurozone==1])
90 # 2.6 Calculate the GDP per capita in euros
91 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
... EU
92 gdp_per_capita<-(Eudata$GDP/Eudata$Population)
93 gdp_per_capita
```

```
94 sum(gdp_per_capita[Eudata$Currency=='euro'])
95 sum(Eudata$GDP[Eudata$Eurozone==1]/Eudata$Population[Eudata$Eurozone==1])
96 sum(Eudata$GDP[Eudata$Eurozone==0]/Eudata$Population[Eudata$Eurozone==0])
97
98 # 2.7 When was the EU founded?
99 # Hint: this must be the earliest year in which any country became a
... member
100 Eudata$Accession[which.min(Eudata$Accession)]
101
102 # 2.8 Calculate the number of EU founding members
103 sum(Eudata$Accession==1953)
104 # 2.9 Only now you discover that the data set still contains the UK.
105 # Permanently remove the UK from the dataframe "Eudata"
106 Eudata<-Eudata[Eudata$CountyName != 'United Kingdom',]
107 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
... January 2026.
108 # Permanently update the dataframe "Eudata" accordingly.
109 Eudata$Accession[3,4]<- 2026
110 #qui devo cercare di sostituire la casella accession nel rigo 3, mettendo
... la data 2026
111
112
113 ######
114 # 3: Simulation and probability (15 points) #
115 #####
116 # 3.1 Use R to produce one roll of a dice.
117 sample(1:6,1)
118 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
119 k<-sample(1:6,1000, replace = TRUE)
120 k
121 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
122 # a "4" or "5" from a dice
123 prob4<-mean(k==4)
124 prob4
125 prob5<-mean(k==5)
126 prob5
127 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
... a dice
128 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
129 # and variance of the random variable z = 2k-m
130 m<- sample(1:6,1000, replace = T)
131 m
132 k
133 z<-2*k-m
134 expectedvalue<-mean(z)
135 round(expectedvalue)
136 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
137 # a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
138 stock<-rnorm(mean=0.12, sd=0.2, 100)
139 stock
```

```
140 # 3.6 What is the probability of a negative stock return?  
141 #     Answer this question ...  
142 #     (a) by using the variable "stock" from (3.5)  
143 #     (b) by calculating the (theoretical) probability for a normal  
... distribution  
144 mean(pnorm(stock<0))  
145 round(mean(pnorm(stock<0)))  
146 #####  
147 # 4: Functions and Optimization      (25 points)  
148 #####  
149 # 4.1 Create the function f(x)=x^2 in R  
150 f<- function(x){  
151   x^2  
152 }  
153  
154 # 4.2 Calculate the value of f for x=1  
155 f(1)  
156 # 4.3 Create a plot of the function for the interval [-2, 2]  
157 #     If in doubt, type "?plot" to get the help file for the function  
158 plot(f,-2,2)  
159 # 4.4 Numerically, by using R, find the location of the minimum using the  
... optim  
160 #     function. Store the result of your minimization (the location of  
... the minimum)  
161 #     in a variable called xmin  
162 xmin<- optim(par=1,fn=f,gr=NULL, method = 'BFGS')  
163 xmin  
164 # 4.5 Now try to find the location of the minimum by implementing a grid  
... search  
165 #     yourself. Choose N=100 grid points. Search in an interval between  
... -2 and 2.  
166 #     Store the result of your minimization (the location of the minimum)  
... in a  
167 #     variable called xmin_grid  
168  
169 #in questo caso ho provato ad utilizzare questo metodo  
... dell'ottimizzazione con grid  
170 #nonostante non riesca a svolgerlo  
171 kgrid<-seq(0,100,1)  
172 ugrid<-c(kgrid)  
173 k_ottimo<- kgrid[which.min(ugrid)]  
174 k_ottimo  
175 xmin_grid<- optim(par=1,fn=kgrid, gr=grad ,lower=-2, upper=2 ,method =  
... 'BFGS')  
176  
177 #in questo caso devo utilizzare il comando grid research per trovare il  
178 #punto di minimo della variabile xmin_grid  
179 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)  
... and (4.5)  
180 #     are not identical. Why?  
181 #4.4 is negative infinity while the 4.5 is between -2 and 2  
182
```

```
183 #####  
184 # 5: Functions and algorithms (25 points) #  
185 #####  
186 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
... is valid. It goes  
187 # like this  
188 # a) process individual digits from right to left  
189 # b) leave digits number 1,3,5 etc (counted from right) unchanged  
190 # c) multiply digits 2,3,6 etc (counted from right) by 2  
191 # d) if a digit (after multiplying by 2) is larger than 9, subtract  
9  
...  
192 # e) calculate the sum of all (processed) digits  
193 # IF the result can be divided by 10, the number is a valid credit  
... card number  
194 #  
195 # Example: 63487  
196 # Right to left: do not change 7, multiply 8 by 2 and subtract 9, do  
... not change 4,  
197 # multiply 3 by 2 and do not subtract anything, do  
... not change 6  
198 # 7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is  
... valid  
199 # Hint: The operation x %% y yields the remainder of the division  
... x/y.  
200 # For instance, 7 %% 4 gives 3  
201  
202 # Write a function called checkLuhn that takes as argument a vector of  
... individual digits  
203 # and returns TRUE if the number is a valid number and FALSE if it is not  
... valid.  
204 checkLuhn<- function(x){  
205   count= 0  
206   if(x==1 && 3 && 5){  
207     return(x)  
208   }  
209   else if(x==2 && 3 && 6){  
210     return(x*2)  
211   }  
212   else if (x*2>9){  
213     return(x-9)  
214   }  
215   return(x)  
216 }  
217  
218 # The following line takes a *valid* credit card number and creates a  
... vector with single digits.  
219 # You can use this to test your function  
220 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn  
... algorithm works from  
221 # right to left.  
222 digits <- as.numeric(unlist(strsplit("5019717010103742","")))  
223
```

224

225

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <Malak El fatih>
31
32
33 ######
34 # 1: R basics      (13 points)          #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 = seq(10,0,-1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 = 3 *(1:10)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 M <- matrix(seq(1, 300, by = 1), nrow = 5, byrow = TRUE)
44 # 1.4 Print the first row of M
45 M[1, ]
46 # 1.5 Display the last element of v1. Tell R to "display the last element
47 ... of x",
48 #      regardless of the dimension of v1.
```

```
47 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.  
48 res <- c((12/(19-7))^^(1/5),  
49     (log(1) + log(2)) / ((pi + 1)/(pi - 1 )),  
50     log(sin(2)/exp(2))  
51 )  
52 # 1.7 Create a variable "u" with the value "ten to the power minus 5"  
53  
54 # 1.8 If you would now run the line  
55 # print(U)  
56 # you would get an error message. Which important concept of R is the  
... reason  
57 # for this error? Answer in 1 sentence as a comment.  
58  
59  
60 #####  
61 # 2: Data and Logical conditions (20 points) #  
62 #####  
63 # The file "Eudata.csv" contains data about the (still 28) EU countries.  
64 # The columns are: County Name, Code, Capital, Accession (=Year of  
... membership),  
65 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if  
... member)  
66  
67 # The following line loads the data into an R dataframe  
68 # Hunt: Use Session/Set Working Directory/To Source File Location  
69 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)  
70  
71 # 2.1 How many countries are there in the dataset?  
72 nrow(Eudata)  
73  
74 # 2.2 Calculate the total population of the EU  
75 sum(Eudata$Population)  
76  
77 # 2.3 Print the population of the smallest and largest EU country by Area  
78  
79 pop_smallest <- Eudata$Population[which.min(Eudata$Area)]  
80  
81 pop_largest <- Eudata$Population[which.max(Eudata$Area)]  
82  
83 pop_smallest  
84 pop_largest  
85  
86 # 2.4 Calculate the number of countries that are members of the Eurozone  
87 sum(Eudata$IsEurozone == 1)  
88  
89 # 2.5 Calculate the total GDP of all Eurozone members  
90 sum(Eudata$GDP[Eudata$IsEurozone == 1])  
91  
92 # 2.6 Calculate the GDP per capita in euros  
93 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone  
... EU  
94 # (a)
```

```
95 gdp_pc_eu <- sum(Eudata$GDP) * 1e6 / sum(Eudata$Population)
96
97 # (b)
98 gdp_pc_euro <- sum(Eudata$GDP[Eudata$IsEurozone == 1]) * 1e6 /
99   sum(Eudata$Population[Eudata$IsEurozone == 1])
100
101 # (c)
102 gdp_pc_noneuro <- sum(Eudata$GDP[Eudata$IsEurozone == 0]) * 1e6 /
103   sum(Eudata$Population[Eudata$IsEurozone == 0])
104
105 gdp_pc_eu
106 gdp_pc_euro
107 gdp_pc_noneuro
108
109 # 2.7 When was the EU founded?
110 # Hint: this must be the earliest year in which any country became a
... member
111
112 # 2.8 Calculate the number of EU founding members
113
114 # 2.9 Only now you discover that the data set still contains the UK.
115 # Permanently remove the UK from the dataframe "Eudata"
116
117 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
... January 2026.
118 # Permanently update the dataframe "Eudata" accordingly.
119
120
121
122 ######
123 # 3: Simulation and probability (15 points) #
124 #####
125 # 3.1 Use R to produce one roll of a dice.
126 sample(1:6, 1)
127
128 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
129 k <- sample(1:6, 1000, replace = TRUE)
130
131 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
132 # a "4" or "5" from a dice
133 mean(k == 4 | k == 5)
134
135 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
... a dice
136 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
137 # and variance of the random variable z = 2k-m
138 m <- sample(1:6, 1000, replace = TRUE)
139
140 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
141 # a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
```

```
142 stock <- rnorm(100, mean = 0.12, sd = 0.2)
143
144 # 3.6 What is the probability of a negative stock return?
145 #     Answer this question ...
146 #     (a) by using the variable "stock" from (3.5)
147 #     (b) by calculating the (theoretical) probability for a normal
148 ... distribution
149 mean(stock < 0)
150 pnorm(0, mean = 0.12, sd = 0.2)
151
152 #####
153 # 4: Functions and Optimization      (25 points)#
154 #####
155 # 4.1 Create the function f(x)=x^2 in R
156 f <- function(x) {
157   x^2
158 }
159
160 # 4.2 Calculate the value of f for x=1
161 f(1)
162 # [1] 1
163
164
165 # 4.3 Create a plot of the function for the interval [-2, 2]
166 #     If in doubt, type "?plot" to get the help file for the function
167 x <- seq(-2, 2, length.out = 400)
168 plot(x, f(x), type = "l", xlab = "x", ylab = "f(x)", main = "f(x) = x^2
169 ... on [-2, 2]")
170
171 # 4.4 Numerically, by using R, find the location of the minimum using the
172 ... optim
173 #     function. Store the result of your minimization (the location of
174 ... the minimum)
175 #     in a variable called xmin
176 xmin <- optimize(g, interval = c(-10,10))$minimum
177
178 # 4.5 Now try to find the location of the minimum by implementing a grid
179 ... search
180 #     yourself. Choose N=100 grid points. Search in an interval between
181 ... -2 and 2.
182 #     Store the result of your minimization (the location of the minimum)
183 ... in a
184 #     variable called xmin_grid
185 N <- 100
186 grid <- seq(-2, 2, length.out = N)
187 vals <- f(grid)
188
189 xmin_grid <- grid[which.min(vals)]
190 xmin_grid
```

```
187 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
188 #      are not identical. Why?
189
190
191
192
193 y#####
194 # 5: Functions and algorithms (25 points) #
195 #####
196 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
197 #      like this
198 #      a) process individual digits from right to left
199 rev_digits <- rev(digits)
200 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
201 #      c) multiply digits 2,3,6 etc (counted from right) by 2
202 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
203 #      e) calculate the sum of all (processed) digits
204 #      IF the result can be divided by 10, the number is a valid credit
... card number
205 #
206 #      Example: 63487
207 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
208 #                      multiply 3 by 2 and do not subtract anything, do
... not change 6
209 #                      7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
210 #      Hint: The operation x %% y yields the remainder of the division
... x/y.
211 #              For instance, 7 %% 4 gives 3
212
213 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
214 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
215
216 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
217 # You can use this to test your function
218 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
219 # right to left.
220 digits <- as.numeric(unlist(strsplit("5019717010103742","","")))
221
222 checkLuhn <- function(digits) {
223   digits <- as.integer(digits)
224   if (any(is.na(digits)) || any(digits < 0 | digits > 9)) return(FALSE)
225
226   a) process individual digits from right to left
```

```
227 rev_digits <- rev(digits)
228
229 #leave digits number 1,3,5 ecc.. (counted from right) unchanged
230 pos <- seq_along(rev_digits)
231
232 # multiply digits 2,3,6 ecc.. (counted from right) by 2
233 to_double <- (pos %% 2 == 0)
234 rev_digits[to_double] <- rev_digits[to_double] * 2
235
236 #if a digit (after multiplying by 2) is larger than 9, subtract 9
237 rev_digits[rev_digits > 9] <- rev_digits[rev_digits > 9] - 9
238
239 # calculate the sum of all (processed) digits
240 sum(rev_digits) %% 10 == 0
241
242
243
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 #####
29 # 0.1 Write your name below as a comment
30 # <Matteo Gangi>
31 # "Matteo Gangi"
32
33
34 ######
35 # 1: R basics      (13 points)          #
36 #####
37 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
38 ... not use the c() command.
39 v1=seq(10, 1, -1)
40
41 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
42 ... not use the c() command.
43 v2= seq(1, 30, 3)
44
45 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
46 M= rbind(seq(1, 291, 10), seq(2, 292, 10), seq(3, 293, 10), seq(4, 294, 10),
... seq(10, 300, 10))
47
48 # 1.4 Print the first row of M
```

```
47 M[1,]  
48  
49 # 1.5 Display the last element of v1. Tell R to "display the last element  
... of x",  
# regardless of the dimension of v1.  
50 tail(v1, 1)  
51  
52  
53 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.  
54 (12/(19-7)) ^ (1/5)  
55 (log10(1) + log0(2))/ ((pi+1)/(pi-1))  
56 log10(sin(2)/exp(2))  
57  
58 # 1.7 Create a variable "u" with the value "ten to the power minus 5"  
59 u= 10^(-5)  
60  
61 # 1.8 If you would now run the line  
62 # print(U)  
63  
64 # you would get an error message. Which important concept of R is the  
... reason  
65 # for this error? Answer in 1 sentence as a comment.  
66  
67 # Because R is case sensitive  
68  
69  
70 #####  
71 # 2: Data and Logical conditions (20 points) #  
72 #####  
73 # The file "Eudata.csv" contains data about the (still 28) EU countries.  
74 # The columns are: County Name, Code, Capital, Accession (=Year of  
... membership),  
75 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if  
... member)  
76  
77 # The following line loads the data into an R dataframe  
78 # Hint: Use Session/Set Working Directory/To Source File Location  
79 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)  
80  
81 # 2.1 How many countries are there in the dataset?  
82 nrow(Eudata)  
83  
84 # 2.2 Calculate the total population of the EU  
85 sum(Eudata$Population)  
86  
87  
88 # 2.3 Print the population of the smallest and largest EU country by Area  
89 smallest= Eudata$Population[which.min(Eudata$Area)]  
90 largest = Eudata$Population [which.max(Eudata$Area)]  
91  
92 # 2.4 Calculate the number of countries that are members of the Eurozone  
93 sum(Eudata$Eurozone== TRUE)  
94
```

```
95 # 2.5 Calculate the total GDP of all Eurozone members
96 sum(Eudata$GDP[Eudata$Eurozone== TRUE])
97
98 # 2.6 Calculate the GDP per capita in euros
99 #     (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
...
EU
100
101 sum(Eudata$GDP/Eudata$Population)           #a
102 sum(Eudata$GDP[Eudata$Eurozone==
... TRUE]/Eudata$Population[Eudata$Eurozone== TRUE])      #b
103 sum(Eudata$GDP[Eudata$Eurozone==
... FALSE]/Eudata$Population[Eudata$Eurozone== FALSE])
104
105 # 2.7 When was the EU founded?
106 #     Hint: this must be the earliest year in which any country became a
...
member
107 min(Eudata$Accession)
108
109 # 2.8 Calculate the number of EU founding members
110 sum(Eudata$Accession == 1953)
111
112 # 2.9 Only now you discover that the data set still contains the UK.
113 #     Permanently remove the UK from the dataframe "Eudata"
114 Eudata= Eudata[Eudata$CountyName != "United Kingdom"]
115
116
117 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
...
January 2026.
118 #     Permanently update the dataframe "Eudata" accordingly.
119
120
121
122 ######
123 # 3: Simulation and probability (15 points) #
124 #####
125 # 3.1 Use R to produce one roll of a dice.
126 dice= sample(1:6, 1, TRUE)
127
128 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
129 k=sample (1:6, 1, TRUE)
130 k=sample (1:6, 1, TRUE)
131
132 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
133 #     a "4" or "5" from a dice
134 mean (k==4 | k==5)
135
136 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
...
a dice
137 #     Using "k" and "m" from (3.2) and (3.4), estimate the expected value
138 #     and variance of the random variable z = 2k-m
139
140 m= sample(1:6, 1000, TRUE)
```

```
141 z= 2*K - m
142 mean(z)
143
144 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
#      a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
145 stock = rnorm(100, 0.12, 0.2)
146
147
148 # 3.6 What is the probability of a negative stock return?
149 #      Answer this question ...
150 #      (a) by using the variable "stock" from (3.5)
151 #      (b) by calculating the (theoretical) probability for a normal
... distribution
152 mean(stock<0)      #a
153 pnorm(0, 0.12, 0.2)      #b
154
155 ######
156 # 4: Functions and Optimization      (25 points)#
157 #####
158 # 4.1 Create the function f(x)=x^2 in R
159 f = function(x){
160   x^2
161 }
162 # 4.2 Calculate the value of f for x=1
163 f(1)
164 # 4.3 Create a plot of the function for the interval [-2, 2]
165 #      If in doubt, type "?plot" to get the help file for the function
166 curve(f, -2, 2)
167
168 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
169 #      function. Store the result of your minimization (the location of
... the minimum)
170 #      in a variable called xmin
171 neg_f = function (x){
172   -f(x)
173 }
174 res= optim(par=0, fn =neg_f, method = "Brent", lower= -2, upper=2)
175 xmin= res$par
176 print(xmin)
177
178 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
179 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
180 #      Store the result of your minimization (the location of the minimum)
... in a
181 #      variable called xmin_grid
182
183
184 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
```

```
184... and (4.5)
185 #      are not identical. Why?
186
187
188 y#####
189 # 5: Functions and algorithms (25 points) #
190 #####
191 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
192 #      like this
193 #      a) process individual digits from right to left
194 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
195 #      c) multiply digits 2,3,6 etc (counted from right) by 2
196 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
197 #      e) calculate the sum of all (processed) digits
198 #          IF the result can be divided by 10, the number is a valid credit
... card number
199 #
200 #      Example: 63487
201 #          Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
202 #                      multiply 3 by 2 and do not subtract anything, do
... not change 6
203 #                      7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
204 #          Hint: The operation x %% y yields the remainder of the division
... x/y.
205 #              For instance, 7 %% 4 gives 3
206
207 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
208 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
209
210 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
211 # You can use this to test your function
212 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
213 # right to left.
214 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <your name>
31 <Matteo Zucchi>
32
33 ######
34 # 1: R basics      (13 points)          #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 <- 10:1
39 v1
40 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
41 ... not use the c() command.
42 v2 <- seq(3, 30, 3)
43 v2
44 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
45 M= rbind(seq(1, 291, 10), seq(2, 292, 10), seq(3, 293, 10), seq(4, 294,
46 ... 10), seq(10, 300, 10))
47 # 1.4 Print the first row of M
48 M[1,]
49 # 1.5 Display the last element of v1. Tell R to "display the last element
```

```
46... of x",
47 #      regardless of the dimension of v1.
48 tail(v1,1)
49 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
50 (12/(19-7))^(1/5)
51 (log10(1)+log10(2))/((pi+1)/(pi-1))
52 log10(sin(2)/exp(2))
53 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
54 u= 10^(-5)
55 # 1.8 If you would now run the line
56 # print(U)
57 # you would get an error message. Which important concept of R is the
...
58 # for this error? Answer in 1 sentence as a comment.
59 --> Because R is case sensitive
60
61 #####
62 # 2: Data and Logical conditions (20 points) #
63 #####
64 # The file "Eudata.csv" contains data about the (still 28) EU countries.
65 # The columns are: County Name, Code, Capital, Accession (=Year of
...
66 # membership),
67 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
...
68 # member)
69
70 # The following line loads the data into an R dataframe
71 # Hint: Use Session/Set Working Directory/To Source File Location
72 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
73
74 # 2.1 How many countries are there in the dataset?
75 nrow(Eudata)
76 # 2.2 Calculate the total population of the EU
77 sum=(Eudata$Population)
78 # 2.3 Print the population of the smallest and largest EU country by Area
79 smallest= Eudata$Population[which.min(Eudata$Area)]
80 largest= Eudata$Population[which.max(Eudata$Area)]
81 # 2.4 Calculate the number of countries that are members of the Eurozone
82 sum(Eudata$Eurozone == TRUE)
83 # 2.5 Calculate the total GDP of all Eurozone members
84 sum(Eudata$GDP[Eudata$Eurozone == TRUE])
85 # 2.6 Calculate the GDP per capita in euros
86 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
...
87 EU
88 sum(Eudata$GDP/Eudata$Population) # (a)
89 sum(Eudata$GDP[Eudata$Eurozone == TRUE]/Eudata$Population[Eudata$Eurozone
...
== TRUE]) #(b)
90 sum(Eudata$GDP[Eudata$Eurozone ==
...
FALSE]/Eudata$Population[Eudata$Eurozone == FALSE]) # (c)
91 # 2.7 When was the EU founded?
92 #      Hint: this must be the earliest year in which any country became a
...
93 member
94 min(Eudata$Accession)
```

```
91 # 2.8 Calculate the number of EU founding members
92 sum(Eudata$Accession == 1953)
93 # 2.9 Only now you discover that the data set still contains the UK.
94 #     Permanently remove the UK from the dataframe "Eudata"
95 Eudata= Eudata[-28,]
96 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
97 ... January 2026.
98 #     Permanently update the dataframe "Eudata" accordingly.
99 Eudata= Eudata[Eudata$CountyName ]
100
101 #####
102 # 3: Simulation and probability (15 points) #
103 #####
104 # 3.1 Use R to produce one roll of a dice.
105 dice= sample(1:6, 1, TRUE)
106 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
107 k= sample(1:6, 1000, TRUE)
108 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
109 #     a "4" or "5" from a dice
110 mean(k == 4 | k == 5)
111 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
112 ... a dice
113 #     Using "k" and "m" from (3.2) and (3.4), estimate the expected value
114 #     and variance of the random variable  $z = 2k - m$ 
115 m= sample(1:6, 1000, TRUE)
116 z= 2*k-m
117 mean(z)
118 # 3.5 Assume the yearly stock return to be normally distributed with a
119 ... mean of 0.12 and
120 #     a standard deviation of 0.2. Create a variable "stock" with 100
121 ... draws of stock returns
122 stock= rnorm(100, 0.12, 0.2)
123 # 3.6 What is the probability of a negative stock return?
124 #     Answer this question ...
125 #     (a) by using the variable "stock" from (3.5)
126 #     (b) by calculating the (theoretical) probability for a normal
127 ... distribution
128 mean(stock<0) #(a)
129 pnorm(0, 0.12, 0.2) #(b)
130
131 #####
132 # 4: Functions and Optimization (25 points)#
133 #####
134 # 4.1 Create the function  $f(x) = x^2$  in R
135 f= function(x){
136   x^2
137 }
138 # 4.2 Calculate the value of f for  $x=1$ 
139 f(1)
140 # 4.3 Create a plot of the function for the interval  $[-2, 2]$ 
141 #     If in doubt, type "?plot" to get the help file for the function
```

```
138 curve(f, -2, 2)
139 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
140 #     function. Store the result of your minimization (the location of
... the minimum)
141 #     in a variable called xmin
142 neg_f= function(x){
143   -f(x)
144 }
145 res= optim(par=0, fn = neg_f,method = "Brent", lower = -2, upper = 2)
146 xmin= res$par
147 print(xmin)
148 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
149 #     yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
150 #     Store the result of your minimization (the location of the minimum)
... in a
151 #     variable called xmin_grid
152 grid()
153 xmin_grid
154 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
155 #     are not identical. Why?
156
157
158 y#####
159 # 5: Functions and algorithms (25 points) #
160 #####
161 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
162 #     like this
163 #     a) process individual digits from right to left
164 #     b) leave digits number 1,3,5 etc (counted from right) unchanged
165 #     c) multiply digits 2,3,6 etc (counted from right) by 2
166 #     d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
167 #     e) calculate the sum of all (processed) digits
168 #         IF the result can be divided by 10, the number is a valid credit
... card number
169 #
170 #         Example: 63487
171 #             Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
172 #                         multiply 3 by 2 and do not subtract anything, do
... not change 6
173 #                         7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
174 #             Hint: The operation x %% y yields the remainder of the division
... x/y.
175 #                 For instance, 7 %% 4 gives 3
176
```

```
177 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
178 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
179
180 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
181 # You can use this to test your function
182 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
183 # right to left.
184 digits <- as.numeric(unlist(strsplit("5019717010103742", "")))
185
186 checkLuhn <- function(x){
187   counter<- 0
188   digits<-
189 }
190
191
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <your name>
31 #<Nicholas Serantoni>
32
33 ######
34 # 1: R basics      (13 points)         #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1<- (10:1)
39 v1
40 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
41 ... not use the c() command.
42 v2<-seq(3,30,by=3)
43 v2
44 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
45 row1<- seq(1,291, by=10)
46 row2<-seq(2,292,by=10)
47 row3<- seq(3,293,by=10)
48 row4<-seq(4,294, by=10)
49 row5<- seq(10,300, by=10)
```

```
48 M<-rbind(row1, row2, row3, row4, row5)
49 M
50 # 1.4 Print the first row of M
51 M["row1",]
52 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
53 #      regardless of the dimension of v1.
54 v1<-(10:1)
55 v1
56 tail(v1,1)
57 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
58 (12/(19-7))^(1/5)
59 (log10(1)+log10(2))/((pi+1)/(pi-1))
60 log10(sin(2)/exp(2))
61 gx<- function(x){
62   g<-3*X^2-x/2+2
63   g
64 }
65 gx
66 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
67 u<-10^5
68 # 1.8 If you would now run the line
69 # print(U)
70 # you would get an error message. Which important concept of R is the
... reason
71 # for this error? Answer in 1 sentence as a comment.
72 #1.8"u"and "U" are seen as two different variables.we should use the same
... lower case name for variables
73
74 ######
75 # 2: Data and Logical conditions (20 points) #
76 #####
77 # The file "Eudata.csv" contains data about the (still 28) EU countries.
78 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
79 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
80
81 # The following line loads the data into an R dataframe
82 # Hint: Use Session/Set Working Directory/To Source File Location
83 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
84 Eudata<- read.csv("~/Desktop/Inf2mock2024 (2)/Eudata.csv")
85 # 2.1 How many countries are there in the dataset?
86 nrow(Eudata)
87 # 2.2 Calculate the total population of the EU
88 sum(Eudata$Population)
89 # 2.3 Print the population of the smallest and largest EU country by Area
90 min(Eudata$Area)
91 max(Eudata$Area)
92 # 2.4 Calculate the number of countries that are members of the Eurozone
93 sum(Eudata$Eurozone)
94 # 2.5 Calculate the total GDP of all Eurozone members
```

```
95 sum(Eudata$GDP[myData$Eurozone])
96 # 2.6 Calculate the GDP per capita in euros
97 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
98 EU ...
99 percepta<-Eudata$GDP/Eudata$Population
100 #a)
101 sum(percepta)
102 #b)
103 sum(percepta(Eudata$Eurozone))
104 # 2.7 When was the EU founded?
105 #      Hint: this must be the earliest year in which any country became a
106 member ...
107 EUfoundation<-min(Eudata$Accession)
108 EUfoundation
109 # 2.8 Calculate the number of EU founding members
110 EUfoundation<-sum(Eudata)
111 # 2.9 Only now you discover that the data set still contains the UK.
112 #      Permanently remove the UK from the dataframe "Eudata"
113 Eudata=Eudata[-28]
114 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
115 January 2026.
116 #      Permanently update the dataframe "Eudata" accordingly.
117 #####
118 # 3: Simulation and probability (15 points) #
119 #####
120 # 3.1 Use R to produce one roll of a dice.
121 sample(1:6,1, replace=T)
122 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
123 k<-sample(1:6,10000,replace =T)
124 k
125 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
126 #      a "4" or "5" from a dice
127 mean(k==4 | k==5)
128 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
129 #      a dice
130 #      Using "k" and "m" from (3.2) and (3.4), estimate the expected value
131 #      and variance of the random variable z = 2k-m
132 m=sample(1:6,10000,TRUE)
133 z=2*k-m
134 mean(z)
135 # 3.5 Assume the yearly stock return to be normally distributed with a
136 #      mean of 0.12 and
137 #      a standard deviation of 0.2. Create a variable "stock" with 100
138 #      draws of stock returns
139 stock<- rnorm(100,mean=0.12,sd=0.2)
140 # 3.6 What is the probability of a negative stock return?
141 #      Answer this question ...
142 #      (a) by using the variable "stock" from (3.5)
143 empirical_probability<-sum(stock<0)/length(stock)
```

```
141 #      (b) by calculating the (theoretical) probability for a normal
142 ... distribution
143 theoretical_probability<-pnorm(0,mean=0.12,sd=0.2)
144 #####
145 # 4: Functions and Optimization      (25 points)#
146 #####
147 # 4.1 Create the function f(x)=x^2 in R
148 fx<- function(x){
149   fx1<- -x^2+1
150   print(fx1)
151 }
152 fx
153 # 4.2 Calculate the value of f for x=1
154 fx(x=1)
155 # 4.3 Create a plot of the function for the interval [-2, 2]
156 #      If in doubt, type "?plot" to get the help file for the function
157 plot(fx,-2,2)
158 # 4.4 Numerically, by using R, find the location of the minimum using the
159 ... optim
160 #      function. Store the result of your minimization (the location of
161 ... the minimum)
162 #      in a variable called xmin
163 xmin<-optimize(fx, interval=c(-2,2))$minimum
164 print(xmin)
165 # 4.5 Now try to find the location of the minimum by implementing a grid
166 ... search
167 #      yourself. Choose N=100 grid points. Search in an interval between
168 ... -2 and 2.
169 #      Store the result of your minimization (the location of the minimum)
170 ... in a
171 #      variable called xmin_grid
172
173 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
174 ... and (4.5)
175 #      are not identical. Why?
176
177
178
179
180
181
182
```

y#####

```
172 y#####
173 # 5: Functions and algorithms      (25 points)      #
174 #####
175 # 5.1 The Luhn Algorithm is used to check whether a credit card number
176 ... is valid. It goes
177 #      like this
178 #      a) process individual digits from right to left
179 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
180 #      c) multiply digits 2,3,6 etc (counted from right) by 2
181 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
182 ... 9
183 #      e) calculate the sum of all (processed) digits
184 #      IF the result can be divided by 10, the number is a valid credit
185 ... card number
```

```
183 #
184 #      Example: 63487
185 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
186 #                      multiply 3 by 2 and do not subtract anything, do
... not change 6
187 #                      7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
188 #      Hint: The operation x %% y yields the remainder of the division
... x/y.
189 #          For instance, 7 %% 4 gives 3
190
191 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
192 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
193 checkLuhn<- function
194 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
195 # You can use this to test your function
196 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
197 # right to left.
198 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <your name>
31
32 # Samuel Boccomino
33
34 ######
35 # 1: R basics      (13 points)          #
36 ######
37 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
38 ... not use the c() command.
39 v1 <- seq(from = 10, to = 1, by = -1)
40 v1
41 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
42 ... not use the c() command.
43 v2 <- seq(from = 3, to = 30, by = 3)
44 v2
45 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
46 r1 <- seq(from = 1, to = 291, by = 10)
47 r2 <- seq(from = 2, to = 292, by = 10)
48 r3 <- seq(from = 3, to = 293, by = 10)
49 r4 <- seq(from = 4, to = 294, by = 10)
```

```
48 r5 <- seq(from = 10, to = 300, by = 10)
49 M <- rbind(r1, r2, r3, r4, r5)
50 M
51 # 1.4 Print the first row of M
52 M[1 ,]
53 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
54 #      regardless of the dimension of v1.
55 tail(v1, 1)
56 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
57 ((12/(19-7)))^(1/5)
58 (log(1)+log(2))/((pi+1)/(pi-1))
59 log(sin(2))/(exp(1))^2
60 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
61 u <- 10^(-5)
62 u
63 # 1.8 If you would now run the line
64 # print(U)
65 # you would get an error message. Which important concept of R is the
... reason
66 # for this error? Answer in 1 sentence as a comment.
67
68 # It doesn't work because R is case sensitive, so U is different from u.
69
70 #####
71 # 2: Data and Logical conditions (20 points) #
72 #####
73 # The file "Eudata.csv" contains data about the (still 28) EU countries.
74 # The columns are: County Name, Code, Capital, Accession (=Year of
... membership),
75 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
... member)
76
77 # The following line loads the data into an R dataframe
78 # Hint: Use Session/Set Working Directory/To Source File Location
79 Eudata <- read.table('Eudata-2.csv', sep=";", header=TRUE)
80 # 2.1 How many countries are there in the dataset?
81 nrow(Eudata)
82 # 2.2 Calculate the total population of the EU
83 sum(Eudata$Population)
84 # 2.3 Print the population of the smallest and largest EU country by Area
85 Eudata$Population[which.min(Eudata$Area)]
86 Eudata$Population[which.max(Eudata$Area)]
87 # 2.4 Calculate the number of countries that are members of the Eurozone
88 sum(Eudata$Eurozone==TRUE)
89 # 2.5 Calculate the total GDP of all Eurozone members
90 sum(Eudata$GDP[Eudata$Eurozone==TRUE])
91 # 2.6 Calculate the GDP per capita in euros
92 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
... EU
93 # (a)
94 total_GDP_euros <- sum(Eudata$GDP)*10^6
```

```
95 total_population <- sum(Eudata$Population)
96 GDP_per_capita_total <- total_GDP_euros / total_population
97 GDP_per_capita_total
98 # (b)
99 GDP_eurozone <- sum(Eudata$GDP[Eudata$Eurozone==TRUE])*10^6
100 pop_eurozone <- sum(Eudata$Population[Eudata$Eurozone==TRUE])
101 GDP_per_capita_eurozone <- GDP_eurozone/ pop_eurozone
102 GDP_per_capita_eurozone
103 # (c)
104 GDP_non_euro <- sum(Eudata$GDP[Eudata$Eurozone==FALSE])*10^6
105 pop_non_euro <- sum(Eudata$Population[Eudata$Eurozone==FALSE])
106 GDP_per_capita_non_euro <- GDP_non_euro / pop_non_euro
107 GDP_per_capita_non_euro
108 # 2.7 When was the EU founded?
109 # Hint: this must be the earliest year in which any country became a
110 ... member
110 min(Eudata$Accession)
111 # 2.8 Calculate the number of EU founding members
112 sum(Eudata$Accession==1953)
113 # 2.9 Only now you discover that the data set still contains the UK.
114 # Permanently remove the UK from the dataframe "Eudata"
115 Eudata <- Eudata[Eudata$CountyName!="United Kingdom", ]
116 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
117 ... January 2026.
117 # Permanently update the dataframe "Eudata" accordingly.
118
119 ##### # 3: Simulation and probability (15 points) #
120 # 3.1 Use R to produce one roll of a dice.
121
122 dice_roll <- sample(1:6,1)
123 dice_roll
124
125 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
126 k <- sample(1:6,1000,replace = TRUE)
127 k
128 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
129 # a "4" or "5" from a dice
130 p4 <- k[k==4]
131 p5 <- k[k==5]
132 probability <- (length(p4)+length(p5))/1000
133 probability
134 percentage <- ((length(p4)+length(p5))/1000)*100
135 percentage
136 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
137 ... a dice
138 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
139 # and variance of the random variable z = 2k-m
139 m <- sample(1:6,1000,replace = TRUE)
140 m
141 z <- 2*k-m
142 z
143 mean(z)
```

```
144 var(z)
145 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
146 #      a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
147 stock <- rnorm(100,mean=0.12,sd=0.2)
148 stock
149 # 3.6 What is the probability of a negative stock return?
150 #      Answer this question ...
151 #      (a) by using the variable "stock" from (3.5)
152 #      (b) by calculating the (theoretical) probability for a normal
... distribution
153
154 # (a)
155 negative_stock_return1 <- sum(stock<0)/length(stock)
156 negative_stock_return1
157 # (b)
158 negative_stock_return2 <- pnorm(0,mean=0.12,sd=0.2)
159 negative_stock_return2
160 #####
161 # 4: Functions and Optimization      (25 points)#
162 #####
163 # 4.1 Create the function f(x)=x^2 in R
164 f <- function(x){x^2}
165 # 4.2 Calculate the value of f for x=1
166 f(1)
167 # 4.3 Create a plot of the function for the interval [-2, 2]
168 #      If in doubt, type "?plot" to get the help file for the function
169 curve(f,-2,2)
170 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
171 #      function. Store the result of your minimization (the location of
... the minimum)
172 #      in a variable called xmin
173 xmin <- optim(-2, f, method = 'L-BFGS-B', lower = -2, upper = 2)
174 xmin
175 # just the par:
176 xmin$par
177 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
178 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
179 #      Store the result of your minimization (the location of the minimum)
... in a
180 #      variable called xmin_grid
181 grid <- seq(from = -2, to = 2, by = 0.04)
182 best_value <- Inf
183 best_x <- NA
184 for(x in grid){
185   current_value <- f(x)
186   if(current_value < best_value){
187     best_value <- current_value
```

```
188     best_x <- x
189   }
190 }
191 xmin_grid <- print(best_value)
192 cat("Minimum of the function:", best_value, "\n")
193 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
194 ... and (4.5)
195 #      are not identical. Why?
196
197 # Because f(x) is a continuos function
198 #####
199 # 5: Functions and algorithms (25 points) #
200 #####
201 # 5.1 The Luhn Algorithm is used to check whether a credit card number
202 ... is valid. It goes
203 #      like this
204 #      a) process individual digits from right to left
205 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
206 #      c) multiply digits 2,3,6 etc (counted from right) by 2
207 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
208 ... 9
209 #      e) calculate the sum of all (processed) digits
210 #      IF the result can be divided by 10, the number is a valid credit
211 ... card number
212 #
213 #      Example: 63487
214 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
215 ... not change 4,
216 #              multiply 3 by 2 and do not subtract anything, do
217 ... not change 6
218 #              7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
219 ... valid
220 #      Hint: The operation x %% y yields the remainder of the division
221 ... x/y.
222 #      For instance, 7 %% 4 gives 3
223
224 # Write a function called checkLuhn that takes as argument a vector of
225 ... individual digits
226 # and returns TRUE if the number is a valid number and FALSE if it is not
227 ... valid.
228
229 checkLuhn <- function(credit_card_number){
230   digits <-
231   as.numeric(unlist(strsplit(as.character(credit_card_number))))
232   n <- length(digits)
233   for(i in seq(n, 1, -2)){
234     if(i>1){
235       digits[i-1] <- digits[i-1]*2
236       if(digits[i-1]>9)
237         digits[i-1] <- digits[i-1]-9
238     }
239 }
```

```
229 }
230     return(sum(digits)%%10==0)
231 }
232 # The following line takes a *valid* credit card number and creates a
233 # vector with single digits.
234 # You can use this to test your function
235 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
236 # algorithm works from
237 # right to left.
238 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
239 digits
```

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
... using R!
6 # - The exam takes 90 minutes and has 5 questions
7 # - There are a total of 100 points
8 # >>>> SUBMIT your answers on iCorsi in time
... <---
9
10 # Material that you can use
11 # - the R help function
12 # - any printed material (open book)
13
14 # Notes
15 # - There are several ways how this exam can be solved.
16 # - What counts are ...
17 #   + that your program works
18 #   + that you follow the instructions
19 #   + that the program fulfills the requirements
20 # - We do not expect you to provide exactly the solution presented in
... class.
21
22 # Tip:
23 # - Use R to check your solution!
24
25
26 ######
27 # 0: Your name (2 points)
28 name<-StefanoLaureti
29 #####
30 # 0.1 Write your name below as a comment
31 # <your name>
32 # <Stefano Laureti>
33
34 #####
35 # 1: R basics (13 points) #
36 #####
37 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
... not use the c() command.
38 v1<-seq(10,1,-1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
... not use the c() command.
40 v2<-seq(3,30, 3)
41 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
42 M <- matrix(seq(1, 300, 2), nrow=5, byrow=FALSE)
43 # 1.4 Print the first row of M
44 M[1, ]
45 # 1.5 Display the last element of v1. Tell R to "display the last element
... of x",
46 #      regardless of the dimension of v1.
```

```
47 tail(v1, 1)
48
49 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
50 calc_1_7 <- c(
51   (12/(19-7))^(1/5),
52   (log10(1) + log10(2)) / ((pi + 1)/(pi - 1)),
53   log10(sin(2) / exp(2)))
54 )
55 calc_1_7
56
57
58 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
59 u<-c(10^-5)
60 u
61
62 # 1.8 If you would now run the line
63 # print(U)
64 # you would get an error message. Which important concept of R is the
...
65 # for this error? Answer in 1 sentence as a comment.
66 # R is case-sensitive: U and u are different objects; also Print() is not
...
67 print().
68
69 #####
70 # 2: Data and Logical conditions (20 points) #
71 #####
72 # The file "Eudata.csv" contains data about the (still 28) EU countries.
73 # The columns are: County Name, Code, Capital, Accession (=Year of
...
74 # membership),
75 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
...
76 # member)
77
78 # The following line loads the data into an R dataframe
79 # Hunt: Use Session/Set Working Directory/To Source File Location
80 Eudata <- read.table('Eudata.csv', sep=";", header=TRUE)
81
82 # 2.1 How many countries are there in the dataset?
83 nrow(Eudata)
84 # 2.2 Calculate the total population of the EU
85 sum(Eudata$Population, na.rm=TRUE)
86 # 2.3 Print the population of the smallest and largest EU country by Area
87 pop_smallest <- Eudata$Population[which.min(Eudata$Area)]
88 pop_largest <- Eudata$Population[which.max(Eudata$Area)]
89 pop_smallest
90 pop_largest
91
92 # 2.4 Calculate the number of countries that are members of the Eurozone
93 sum(Eudata$IsEurozone == 1, na.rm = TRUE)
94 # 2.5 Calculate the total GDP of all Eurozone members
95 sum(Eudata$GDP[Eudata$IsEurozone == 1], na.rm = TRUE)
96 # 2.6 Calculate the GDP per capita in euros
```

```
95 #      (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
96 ... EU
97 gdp_pc_all <- (sum(Eudata$GDP, na.rm = TRUE) * 1e6) /
98 ... sum(Eudata$Population, na.rm = TRUE)
99
100 gdp_pc_ez <- (sum(Eudata$GDP[Eudata$IsEurozone == 1], na.rm = TRUE) *
101 ... 1e6) /
102     sum(Eudata$Population[Eudata$IsEurozone == 1], na.rm = TRUE)
103
104 gdp_pc_nonez <- (sum(Eudata$GDP[Eudata$IsEurozone == 0], na.rm = TRUE) *
105 ... 1e6) /
106     sum(Eudata$Population[Eudata$IsEurozone == 0], na.rm = TRUE)
107
108 gdp_pc_all
109 gdp_pc_ez
110 gdp_pc_nonez
111 # 2.7 When was the EU founded?
112 #      Hint: this must be the earliest year in which any country became a
113 ... member
114 min(Eudata$Accession, na.rm = TRUE)
115 # 2.8 Calculate the number of EU founding members
116 sum(Eudata$Accession == 1953, na.rm = TRUE)
117 # 2.9 Only now you discover that the data set still contains the UK.
118 #      Permanently remove the UK from the dataframe "Eudata"
119 Eudata <- Eudata[Eudata$`County Name` != "United Kingdom", ]
120 # (se il nome è diverso, controllo con: names(Eudata))
121 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
122 ... January 2026.
123 #      Permanently update the dataframe "Eudata" accordingly.
124
125 #####
126 # 3: Simulation and probability (15 points) #
127 #####
128 # 3.1 Use R to produce one roll of a dice.
129 rolldice<-sample(1:6,1,replace=TRUE)
130 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
131 k<-seq(sample(1:6,5000,replace=TRUE))
132 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
133 #      a "4" or "5" from a dice
134 prob<- sum(k%%4)
135 prob_4<-prob/length(k)
136 prob2<- sum(k%%5)
137 prob_5<-prob2/length(k)
138 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
139 ... a dice
140 #      Using "k" and "m" from (3.2) and (3.4), estimate the expected value
141 #      and variance of the random variable z = 2k-m
142 m<-seq(sample(1:6,1000,replace=TRUE))
143 z<-c(2*k-m)
144 z
```

```
140 # 3.5 Assume the yearly stock return to be normally distributed with a
... mean of 0.12 and
141 #      a standard deviation of 0.2. Create a variable "stock" with 100
... draws of stock returns
142 mu<-0.12
143 sdr=sqrt(0.2)
144 stock100<-rnorm(100, mean=mu, sd=sdr)
145 sum(stock100>0)
146 # 3.6 What is the probability of a negative stock return?
147 #      Answer this question ...
148 #      (a) by using the variable "stock" from (3.5)
149 pnorm(0, mean=mu, sd=sdr)
150 #      (b) by calculating the (theoretical) probability for a normal
... distribution
151 mean(pnorm(0, mean=mu, sd=sdr)<0)

152
153
154 ######
155 # 4: Functions and Optimization      (25 points)#
156 #####
157 # 4.1 Create the function f(x)=x^2 in R
158 f<-function(x) x^2
159 # 4.2 Calculate the value of f for x=1
160 f(1)
161 # 4.3 Create a plot of the function for the interval [-2, 2]
162 #      If in doubt, type "?plot" to get the help file for the function
163 curve(f, from = -2, to = 2)
164 # 4.4 Numerically, by using R, find the location of the minimum using the
... optim
165 #      function. Store the result of your minimization (the location of
... the minimum)
166 #      in a variable called xmin
167 xmin <- optimize(f, interval = c(-2, 2))$minimum
168 xmin
169 f(xmin)
170 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
171 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
172 #      Store the result of your minimization (the location of the minimum)
... in a
173 #      variable called xmin_grid
174 xmin_grid <- optimize(f, interval = c(-2, 2), n=100)$minimum
175 xmin_grid
176 f(xmin_grid)
177 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
178 #      are not identical. Why?
179
180 # Because x^2 -> -Inf as |x| -> Inf, so the function is unbounded below.
181 #####
182 # 5: Functions and algorithms      (25 points)      #
```

```
183 #####  
184 # 5.1 The Luhn Algorithm is used to check whether a credit card number  
... is valid. It goes  
185 # like this  
186 # a) process individual digits from right to left  
187 # b) leave digits number 1,3,5 etc (counted from right) unchanged  
188 # c) multiply digits 2,3,6 etc (counted from right) by 2  
189 # d) if a digit (after multiplying by 2) is larger than 9, subtract  
... 9  
190 # e) calculate the sum of all (processed) digits  
191 # IF the result can be divided by 10, the number is a valid credit  
... card number  
192 # if (n %% 2 > 9) { n <- n-9 }  
193  
194  
195 Luhn<-function(n) { if (x>=0 & x<9){  
196 }  
197 }  
198 }  
199 }  
200  
201 if (n %% 2 > 9) { n <- n-9  
202 }  
203 while (n != 1) {  
204   n <- if (n %% 2 == 0) n/2 else 3*n + 1  
205   out <- c(out, n)  
206 }  
207 #  
208 # Example: 63487  
209 # Right to left: do not change 7, multiply 8 by 2 and subtract 9, do  
... not change 4,  
210 # multiply 3 by 2 and do not subtract anything, do  
... not change 6  
211 # 7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is  
... valid  
212 # Hint: The operation x %% y yields the remainder of the division  
... x/y.  
213 # For instance, 7 %% 4 gives 3  
214  
215 # Write a function called checkLuhn that takes as argument a vector of  
... individual digits  
216 # and returns TRUE if the number is a valid number and FALSE if it is not  
... valid.  
217  
218 # The following line takes a *valid* credit card number and creates a  
... vector with single digits.  
219 # You can use this to test your function  
220 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn  
... algorithm works from  
221 # right to left.  
222 digits <- as.numeric(unlist(strsplit("5019717010103742","")))  
223
```

224

225

```
1 # Exam "Informatica 2", 2025-09-04
2 # Peter Gruber and Paul Schneider
3
4 # Rules
5 # - Except for theoretical questions, all questions must be answered
6 ... using R!
7 # - The exam takes 90 minutes and has 5 questions
8 # - There are a total of 100 points
9 # >>>> SUBMIT your answers on iCorsi in time
10 ... IMPORTANT!!! <----
11
12 # Material that you can use
13 # - the R help function
14 # - any printed material (open book)
15
16 # Notes
17 # - There are several ways how this exam can be solved.
18 # - What counts are ...
19 #   + that your program works
20 #   + that you follow the instructions
21 #   + that the program fulfills the requirements
22 # - We do not expect you to provide exactly the solution presented in
23 ... class.
24
25
26 ######
27 # 0: Your name      (2 points)          #
28 ######
29 # 0.1 Write your name below as a comment
30 # <Mokabbal Assiya>
31
32
33 ######
34 # 1: R basics      (13 points)         #
35 ######
36 # 1.1 Create a vector v1 as defined in equation 1.1 on the Info Sheet. Do
37 ... not use the c() command.
38 v1 = seq(10, 1, -1)
39 # 1.2 Create a vector v2 as defined in equation 1.2 on the Info Sheet. Do
40 ... not use the c() command.
41 v2 = seq(3, 30, 3)
42 # 1.3 Create a matrix M as defined in equation 1.3 on the Info Sheet.
43 M = rbind(seq(1, 291, 10), seq(2, 292, 10), seq(3, 292, 10), seq(4, 294,
44 ... 10), seq(10, 300, 10))
45 # 1.4 Print the first row of M
46 M[1,]
47 # 1.5 Display the last element of v1. Tell R to "display the last element
48 ... of x",
49 #      regardless of the dimension of v1.
```

```
46 tail(v1, 1)
47 # 1.6 Perform the calculations in equation 1.4 on the Info Sheet.
48 (12-(19-7))^1/5
49 log(1)+log(2)/((pi+1)/(pi-1))
50 log(sin(2)/exp(2))
51 # 1.7 Create a variable "u" with the value "ten to the power minus 5"
52 u = 10^(-5)
53 # 1.8 If you would now run the line
54 # print(U)
55 # you would get an error message. Which important concept of R is the
56 ... reason
57 # for this error? Answer in 1 sentence as a comment.
58
59 "R is case sensitive"
60 ######
61 # 2: Data and Logical conditions (20 points) #
62 ######
63 # The file "Eudata.csv" contains data about the (still 28) EU countries.
64 # The columns are: County Name, Code, Capital, Accession (=Year of
65 ... membership),
66 # Population, Area, GDP (in Million EUR), currency, IsEurozone (=1, if
67 ... member)
68
69
70 # 2.1 How many countries are there in the dataset?
71 nrow(Eudata)
72 # 2.2 Calculate the total population of the EU
73 sum(Eudata$Population)
74 # 2.3 Print the population of the smallest and largest EU country by Area
75 smallest= Eudata$Population[which.min(Eudata$Area)]
76 largest= Eudata$Population[which.max(Eudata$Area)]
77 # 2.4 Calculate the number of countries that are members of the Eurozone
78 sum(Eudata$Eurozone==TRUE)
79 # 2.5 Calculate the total GDP of all Eurozone members
80 sum(Eudata$GDP[Eudata$Eurozone == TRUE])
81 # 2.6 Calculate the GDP per capita in euros
82 #     (a) of the total EU, (b) of the Eurozone (c) of the non-Eurozone
83 EU
84
85 GDP_per_capita_EU = sum(Eudata$GDP/Eudata$Population)
86 GDP_Eurozone =
87 ... sum(Eudata$GDP[Eudata$Eurozone]/Eudata$Population[Eudata$Eurozone])
88 GDP_non_EU =
89 ... sum(Eudata$GDP[Eudata$Eurozone==FALSE]/Eudata$Population[Eudata$Eurozone=
90 ... =FALSE])
91
92
93 # 2.7 When was the EU founded?
94 #     Hint: this must be the earliest year in which any country became a
95 ... member
```

```
90 min(Eudata$Accession)
91 # 2.8 Calculate the number of EU founding members
92 sum(Eudata$Accession == 1953)
93 # 2.9 Only now you discover that the data set still contains the UK.
94 # Permanently remove the UK from the dataframe "Eudata"
95 Eudata = Eudata[Eudata$CountyName != "United Kingdom", ]
96 # 2.10 You also discover that Bulgaria actually joined the Euro on 1st
97 ... January 2026.
98 # Permanently update the dataframe "Eudata" accordingly.
99 Eudata = Eudata[Eudata$Eurozone == "Bulgaria"]
100
101 #####
102 # 3: Simulation and probability (15 points) #
103 #####
104 # 3.1 Use R to produce one roll of a dice.
105 dice = sample(1:6, 1, replace= TRUE)
106 # 3.2 Create a vector called "k" that contains 1000 rolls of a dice
107 k= sample(1:6, 1000, replace = TRUE)
108 # 3.3 Using "k" from (3.2), estimate the probability of obtaining
109 # a "4" or "5" from a dice
110 mean(k==4 | k==5)
111 # 3.4 Create a vector called "m" that contains 1000 (different) rolls of
112 ... a dice
113 # Using "k" and "m" from (3.2) and (3.4), estimate the expected value
114 # and variance of the random variable z = 2k-m
115 m = sample(1:6, 1000, replace = TRUE)
116 z = 2*k-m
117 mean(z)
118 # 3.5 Assume the yearly stock return to be normally distributed with a
119 ... mean of 0.12 and
120 # a standard deviation of 0.2. Create a variable "stock" with 100
121 ... draws of stock returns
122 returns_stock = rnorm(100, 0.12, 0.2)
123 # 3.6 What is the probability of a negative stock return?
124 # Answer this question ...
125 # (a) by using the variable "stock" from (3.5)
126 # (b) by calculating the (theoretical) probability for a normal
127 ... distribution
128 mean(returns_stock > 0)
129 pnorm(0, 0.12, 0.2)
130 #####
131 # 4: Functions and Optimization (25 points)#
132 #####
133 # 4.1 Create the function f(x)=x^2 in R
134 f = function(x){x^2}
135 # 4.2 Calculate the value of f for x=1
136 f(1)
137 # 4.3 Create a plot of the function for the interval [-2, 2]
138 # If in doubt, type "?plot" to get the help file for the function
139 curve(f, -2, 2)
140 # 4.4 Numerically, by using R, find the location of the minimum using the
```

```
136... optim
137 #      function. Store the result of your minimization (the location of
... the minimum)
138 #      in a variable called xmin
139 neg_f = function(x){
140   -f(x)
141 }
142 res <- optim(par=0, fn= neg_f, method = "Brent", lower = -2, upper = 2)
143 xmin <- res$par
144 # 4.5 Now try to find the location of the minimum by implementing a grid
... search
145 #      yourself. Choose N=100 grid points. Search in an interval between
... -2 and 2.
146 #      Store the result of your minimization (the location of the minimum)
... in a
147 #      variable called xmin_grid
148
149 # 4.6 Answer in a short comment (<= 2 sentences). The results from (4.4)
... and (4.5)
150 #      are not identical. Why?
151
152
153 y#####
154 # 5: Functions and algorithms (25 points) #
155 #####
156 # 5.1 The Luhn Algorithm is used to check whether a credit card number
... is valid. It goes
157 #      like this
158 #      a) process individual digits from right to left
159 #      b) leave digits number 1,3,5 etc (counted from right) unchanged
160 #      c) multiply digits 2,3,6 etc (counted from right) by 2
161 #      d) if a digit (after multiplying by 2) is larger than 9, subtract
... 9
162 #      e) calculate the sum of all (processed) digits
163 #      IF the result can be divided by 10, the number is a valid credit
... card number
164 #
165 #      Example: 63487
166 #      Right to left: do not change 7, multiply 8 by 2 and subtract 9, do
... not change 4,
167 #                      multiply 3 by 2 and do not subtract anything, do
... not change 6
168 #                      7 + 8*2-9 + 4 + 3*2 + 6 = 30 --> the number is
... valid
169 #      Hint: The operation x %% y yields the remainder of the division
... x/y.
170 #              For instance, 7 %% 4 gives 3
171
172 # Write a function called checkLuhn that takes as argument a vector of
... individual digits
173 # and returns TRUE if the number is a valid number and FALSE if it is not
... valid.
```

```
174
175 # The following line takes a *valid* credit card number and creates a
... vector with single digits.
176 # You can use this to test your function
177 # Hint: digits contains the numbers from LEFT to RIGHT, while the Luhn
... algorithm works from
178 # right to left.
179 digits <- as.numeric(unlist(strsplit("5019717010103742","")))
180
181
182
```