Examination Advanced R Programming

Linköpings Universitet, IDA, Statistik

Course code and name:	732A94 Advanced R Programming
Date:	2020/10/22, 14–18
Teacher:	Krzysztof Bartoszek
Allowed aids:	The extra material is included in the zip file **exam_material_732A94.zip**
Grades:	A= $[18-20]$ points B= $[16-18)$ points C= $[14-16)$ points D= $[12-14)$ points E= $[10-12)$ points F= $[0-10)$ points
Instructions:	Write your answers in an R script file named **[your exam account].R** The R code should be complete and readable code, possible to run by copying directly into a script. Comment directly in the code whenever something needs to be explained or discussed. Follow the instructions carefully. There are **THREE** problems (with sub–questions) to solve. If you also need to provide some hand–written derivations please number each page according to the pattern: Question number . page in question number i.e. Q1.1, Q1.2, Q1.3,. . ., Q2.1, Q2.2, . . ., Q3.1, ... Scan/take photos of such derivations preferably into a single pdf file but if this is not possible multiple pdf or .bmp/.jpg/.png files are fine. Please do not use other formats for scanned/photographed solutions. Please submit all your solutions via LISAM or e–mail. If emailing, please email them to **BOTH** krzysztof.bartoszek@liu.se and KB_LiU_exam@protonmail.ch . During the exam you may ask the examiner questions by emailing them to KB_LiU_exam@protonmail.ch **ONLY**. Other exam procedures in LISAM.

Problem 1 (5p)

a) (2p) Describe how one makes S3 methods implemented in a package for some class available for the user of the package.
b) (3p) What is the Depends field for in the DESCRIPTION file of a `R` package? Why does CRAN not allow much to be listed there?

Problem 2 (10p)

READ THE WHOLE QUESTION BEFORE STARTING TO IMPLEMENT! Remember that your functions should **ALWAYS** check for correctness of user input!

a) (4p) In this task you should use object oriented programming in S3 or RC to write code that simulates a robot vacuum cleaner that cleans a square room. The vacuum cleaner object has to contain information on the robot's current status, how dirty the room is and a mechanism to move the robot to dirty parts. The room is a square of size $n \times n$ tiles. The robot is able to vacuum a whole tile at one go. Each tile is defined by a pair of integers (i, j), where $1 \leq i \leq n$ and $1 \leq j \leq n$.

Your goal is to first construct the robot with a map of the room. Depending on your chosen OO system you can do it through a constructor or by implementing a function `create_robot_cleaner()`. The constructing function should take two arguments—the size of the room (a single integer) and the initial location of the robot in the room (a pair of integers). The robot object should contain information on each tile in the room, whether it is clean or dirty, how many dirty tiles there are, the current location of the robot and the battery status of the robot. You may initially choose which tiles are dirty in any way you like (random, deterministic, your choice). However at least $n^2/4$ have to be dirty and at least $n^2/4$ have to be clean. The initial battery status of the robot should be 500. Provide some example calls to your code.

```
## example call to create_robot_cleaner() function
robot_cleaner <- create_robot_cleaner(n=10,current_location=c(1,1))
```

b) (4p) Now implement a function called `go_clean_tile()`. The function should not take any parameters. The function is to find the closest, in terms of moves of the robot, dirty tile. The robot can move only left, right, up and down. Each such move (left, right, up, down) decreases the robot's battery status by 1. The battery status and location have to be updated after each move. If the robot runs out of battery, then the function has to end with some appropriate comment to the user. After reaching the designated tile, the robot cleans it and the status of the tile has to be changed to clean (do not forget about the counter of the dirty tiles). If when the function is called, there are no dirty tiles, then an appropriate message has to be provided to the user. Provide some example calls to your code.

```
## S3 and RC call
robot_cleaner <-go_clean_tile(robot_cleaner)

## if using RC you may also call in this way
robot_cleaner$go_clean_tile()
```

c) (2p) Implement a function that displays the state of the room. You are free to choose yourself how to report the state! This function has to also work with `print()`.

```
# call to show state of room
robot_cleaner; print(robot_cleaner)
```

Problem 3 (5p)

a) (2p) Binary (that have entries equal to either 0 or 1) matrices show up very often when modelling real world phenomena. It is sometimes important to know how many 1s are present in such a matrix. Implement a function that takes as its input a binary matrix and returns the number of 1s in it. Your implementation has to be done using `for` loops. In particular you may not use any `R` functionality like `sum()` or similar. Do not forget to check for correctness of input.
b) (1p) What is the computational complexity in terms of the matrix's size, i.e. number of rows and columns?
c) (2p) Implement a unit test that compares your implementation with direct R calculations, i.e. `sum(M)`, where `M` is the matrix.