Examination Advanced R Programming

Linköpings Universitet, IDA, Statistik

Course code and name:	732A94 Advanced R Programming
Date:	2021/03/03, 8–12
Teacher:	Krzysztof Bartoszek
Allowed aids:	The extra material is included in the zip file **exam_material.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
	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) In a few sentences provide guidelines on how one should go about optimizing one's program.
b) (3p) Provide examples of at least three programming ways to optimize `R` code.

Problem 2 (10p)

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

a) (3p) In this task you should use object oriented programming in S3 or RC to write code that simulates an automatic memory management. The memory manager has to contain information on each memory cell, whether it is reserved, if so by what process (assume that each process is indexed by a unique integer) and if that particular process is active or not. The memory is a vector of length n. The memory manager is able to access each cell directly.

Your goal is to first construct the memory manager with a map of the memory. Initially all memory cells are unallocated. Depending on your chosen OO system you can do it through a constructor (RC) or by implementing a function `create_memory_manager()` (S3). The constructing function should take three arguments—the size of the memory (an integer, `n`), the fraction of taken–up memory when the memory manager should check if something can freed (a value in $[0, 1]$, `trigger_check`), and the maximum number of processes that exist in parallel. Provide some example calls to your code.

```
## example call to create a memory_manager object
memory_manager <- create_memory_manager(n=100,trigger_check=0.5,max_processes=1000) # S3
my_memory_manager <- memory_manager$new(n=100,trigger_check=0.5,max_processes=1000) # RC
```

b) (3p) Now implement a function called `new_process()`. The function should take one argument, the amount of memory cells that the process requires. The function should first check if the process can be created, i.e. the maximum number of processes is not reached, then if the amount of memory is available. If any of the two fails, then an appropriate message has to be provided to the user. Otherwise, the process is to be created (it should be assigned a unique integer process id that has to be stored somehow) and memory reserved for that particular process. If after creation the percentage of taken up memory exceeds the value pf `trigger_check`, then garbage collection should be performed, i.e. the memory manager should check if any memory can be freed and if so it should be done. Provide some example calls to your code.

```
## S3 and RC call
memory_manager <-new_process(memory_manager,required_memory=10)

## if using RC you may also call in this way
memory_manager$new_process(required_memory=10)
```

c) (2p) Implement a function that kills a process. The function should take one argument—the integer process id. of the process. If there is no such process the memory manager should return inform the user that there is no such process with that particular id. Otherwise the process should be killed (you decide how). However, no memory should be released.
d) (2p) Implement a function that displays the state of the memory. You are free to choose yourself how to report the state! This function has to also work directly with `print()`.

```
# calls to show state of memory
memory_manager; print(memory_manager)
```

Problem 3 (5p)

a) (2p) Sometimes it is important in a numerical vector to know the position of its order statistics, i.e. which element is the smallest, second smallest, . . ., largest, i.e. what `R`'s `order()` function does. Implement a function that for an integer vector returns the positions of its order statistics. You may not use `R`'s `order()`, `sort()` nor similar functions. Provide example calls to your code. Do not forget to check for correctness of input.
b) (1p) What is the computational complexity of your implementation in terms of the length or the input vector.
c) (2p) Implement a unit test that compares your implementation with the `R` function `order()` that does the same.