# Hackathon 4 ( Feb 25, 2022)

## General Instructions:

**Rules**:

- The allowed libraries are stdio.h and stdlib.h (only for malloc, calloc, realloc, free).
- Your program should be modular. Do not write your entire program in main. Create suitable functions.
- For each function, leave a short comment above it describing what the function does.
- You are not allowed to use variable length arrays (VLA). All dynamic memory allocation must be on the heap.
- Your program should not have memory leaks. Free all heap memory used.
- Your program should take input till EOF. You can detect this by checking if scanf returned -1.

**Deadline:** 1800hrs on Sunday 27th Feb.

**Submission:** On Autojudge. Limit of 3, 3, and 4 submissions for problems 1,2, and 3 respectively. Autojudge will not evaluate problem 1.

### Problem 1

(20 marks)
**Input:**

- A number $k \in \mathbb{N}$ and a string "inputFileName" as command line arguments.
- Zero or more `\n` characters given as input via `stdin`.

**Output:**
Let inputFile denote the file with filename "inputFileName".

- First, output the first $k$ lines of the inputFile.
- Then for each `\n` given as input, output another line of the file.

**Termination:** Your program should terminate when either the end of inputFile has been reached, or the input read is EOF.

**Assumption:** You can assume that $k$ will fit inside `int` on the machine that we will use to run your program.

### Problem 2

(40 marks)
See the attached file for the description of a command provided in "man page" style. Implement the command in C.

Remark: The command is a simpler version of `grep -n`. You can play with `grep -n` and optional arguments `-v` and `-i` to understand exactly what we want you to implement.

## Problem 3

(40 marks)
**Queues:**
A queue is a "first in first out" data structure. Inserting elements is called 'enqueuing'. Removing/deleting elements from a queue is called 'dequeueing'.

If you enqueue (insert) elements $a_1, a_2, \ldots, a_n$ in this order, then when you call dequeue, the first element removed is $a_1$. The second element would be $a_2$ and so on.

**Input:**
Each input line will look like one of the following:

- E $n$`\n`
- D`\n`

where $n$ is a number that will fit inside `int` on the server.

**Goal:**
Maintain a queue $Q$ of elements. Enqueue elements when input starts with "E". Dequeue when the input line is "D".

**Output:**

- If the input line said "E $n$", then enqueue (insert) the number $n$ into $Q$. No output!
- If the input line said "D", then dequeue from $Q$ and print the number that was dequeued.
- If a dequeue was requested on an empty queue, print "Empty".

Terminate all your output lines with `\n`.

**Implementation Rules:**

- You must use a struct to maintain your queue.
- Use convenient typedef(s).
- Create supporting functions.

Although in practice, you would create separate header file(s), we require you to put everything into just one .c file so that our evaluation server can be used.