PWr Week 6
*Algorithms and Data Structures*
Laboratory – **List 6**

1. Write a program with below operations for a **heap**. The heap will be of elements of type int.
    a. `void init(Heap& h, int size)` which initialize empty heap with place for maximum size cells.
    b. `void loadAndMakeHeap(Heap &h, int howMany)` which load from next lines howMany integers (separated by spaces). After loading this function execute heapAdjustment function to created a correct heap.
    c. `void add(Heap &h, int value)` which add new value to the end of array and fix the heap.
    d. `void makeSorted(Heap& h)` which sort the heap. After this operation this 'heap' is not heap but sorted sequence so, it cannot be used add(…) function on it.
    e. `void show(Heap& h)` which show the state of the heap/array

Format of a stream on judgment system is presented in appendix 1.

For **10 points** present solutions for this list till **Week 7**.
For 8 **points** present solutions for this list till **Week 8**.
For 5 **points** present solutions for this list till **Week 9**.
**After Week 9 the list is closed.**

## Appendix 1

The solution will be automated tested with tests from console of presented below format. The test assumes, that there are up to X different heaps, which there are created as the first operation in the test. Each heap can be initialized separately.

If a line is empty or starts from `'#'` sign, the line have to be ignored.
In any other case, your program should print an exclamation mark and write (copy) introduced a line and then, depending on the command follow the correct procedure / function.

If a line has a format:
`GO` *n*
your program has to create *n* heaps (without initialization). The heaps are numbered from 0 like an array of heaps. Default current heap is a heap with the number 0. This operation will be called once as the first command.

If a line has a format:
`CH` *n*
your program has to choose a heap of a number `n`, and all next functions will operate on this heap. There is *n>=0*.

If a line has a format:
`IN v`
your program has to call `init(h,v)` for current heap `h`. For any heap this operation will be called once, before using the heap.

If a line has a format:
`LD n`
your program has to call `loadAndMakeHeap(l,n)` for current heap `h`. Next lines will have n numbers separated by spaces or newlines.

If a line has a format:
`MS`
your program has to call `makeSorted(h)` for current heap `h`.

If a line has a format:
`SH`
your program has to call `show(h)` for current heap `h`.

If a line has a format:
`AD v`
your program has to call `add(h, v)` for current heap `h`.

If a line has a format:
`HA`
your program has to end the execution, writing as the last line “END OF EXECUTION”. Every test ends with this line.

For example for input test:

```
go 3
in 10
ld 10
55 85 77 15 62 20 57 43 12 50
sh
ms
sh
ch 1
in 10
ld 7
9 5 8 3 1 7 4
sh
ad 10
sh
ad 6
sh
ms
sh
ha
```

The output have to be:

```
START
!go 3
!in 10
!ld 10
!sh
85,62,77,43,55,20,57,15,12,50,
!ms
!sh
12,15,20,43,50,55,57,62,77,85,
!ch 1
!in 10
!ld 7
!sh
9,5,8,3,1,7,4,
!ad 10
!sh
10,9,8,5,1,7,4,3,
!ad 6
!sh
10,9,8,6,1,7,4,3,5,
!ms
!sh
1,3,4,5,6,7,8,9,10,
!ha
END OF EXECUTION
```