CMSC 401 – Fall 2018

Programming Assignment 1

(due Wed, 9/12 - 11:59pm)

Dr. Eyuphan Bulut

CMSC 401- Algorithm Analysis with Advanced Data Structures

VCU

School of Engineering | Computer Science

Assignment 1

- Implement Majority-Element finding using Algorithm V from lecture slides
  - Input:
    - array of N positive integers - size of the problem is N
  - Output:
    - 1) **majority element (M.E.)** – element occurring more than N/2 times (order of elements doesn't matter), if M.E. exists
    - 2) **-1,** if the majority element does not exist in the array
- Input should be read into array of integers: int[] (do not use ArrayList)
- The code should work on that array, without re-formatting the data e.g. into a linked list or any other data structure
- The algorithm should have **O(N) time complexity**
- Use of any Java built-in functions/classes is NOT allowed
  - With the exception of functions from Scanner, System.in and System.out (or equivalent) for input/output

Input-output formats

- Input Format:
  - First line: a single integer number N>=1, N<=1,000,000
  - **Following N lines:** each contains a single positive integer containing the elements of the array
    - Each integer will be <=1,000,000,000
  - Input will always be correct w.r.t. the specification above

- Output format:
  - A single line, with a single integer:
    - -1 if the input array has no majority element
    - X if integer X is the majority element of the input array

Input 1:

```
5
1
100
100
1
100
```

Input 2:

```
4
10000
2
10000
3
```

Output 1:

```
100
```

Output 2:

```
-1
```

Algorithm V (from lecture)

- Observation: some pair of elements can be deleted from A without changing M.E. (if M.E. exists)
  - Example:

3	3	4	7	7	7	7	7	7	7	8

- Possible pairs and corresponding deletion effects:
  - (non-M.E., non-M.E.) -> result doesn't change
  - (M.E., non-M.E.) -> result doesn't change
  - (M.E., M.E.)-> result can change

  assuming M.E. exists
- Order in which pairs are deleted is not important

- But we don't know which element is M.E.!

Algorithm V (from lecture)

- Solution:
  - Instead of Delete everything except (M.E., M.E.)
  - Do Delete everything except (X,X)
- Move through the elements from left to right
- Keep deleting pairs of no-equal elements at every opportunity
- M.E. will be the only one remaining
  - Easy implementation with double-linked list
    - Example:

3	7	3	7	7	4	7	7	7	7	8

  - But we can also do these concepts for an array. How?

3	3	3	7	7	4	7	7	7	7	8

Algorithm V (from lecture)

- Array approach:
  - Assign a candidate-M.E. (c-ME) (init:1st element)
  - As you sweep the array,
    - do not delete c-ME, move forward
    - delete a non-c-ME with one c-ME.
    - If another element becomes most frequent in the array so far (i.e., a new c-ME), all instances of the previous one have already been deleted
  - One **approach:** keep current position and keep count of c-ME and decrease upon delete
    - Location of c-MEs is not important, only their count

Algorithm V (from lecture)

- **Boyer & Moore approach**
  - Select first element as candidate M.E., set counter to 1
  - Move forward through the array. When we see:
    - another copy of the c-ME -> increase counter
    - some other element -> decrease counter
  - If the counter becomes 0, assign a new c-ME at current index
    - previous c-ME lost the chance
  - Once the entire array is traversed, c-ME will be the only element left (if exists)

Input/output in Java

- Use Standard I/O to read input and write the result
- For Java, input: System.in, output: System.out
- **"Do Not"s**
  - Do not read from a disk file/write to disk file
  - Do not write anything to screen except the result
    - Ex: Human centric messages ("the result is", "please enter..")
    - Automated grading via script will be used for checking correctness of your output

Submission

- Date due: Wed, Sept 12th, 11:59 pm

- Submission through Blackboard
  - Submit as a zip file with name "1_LastName_FirstName.zip" and include:
    - Java source code in a single file cmsc401.java **(all lower case letters!)**
    - The file should have *your name* in a comment in the first line
    - Remember: in Java, class name should match the file name, and is case sensitive

- Please do NOT create your own packages
- Do NOT place the file into a folder – just zip the file
- Use standard I/O to read input (System.in, System.out) and output
- Make sure the **program compiles and WORKS!**
- Late submissions are accepted up to 2 days only!