1. Exam II moved to ELRC 1100
2. Presentation:
Deadline: November 30 ---> December 01
5 slides (8 - 10 min):
1. Introduction; 2. Technologies; 3. Implementation (video demo)
4. Who did what?; 5. Conclusion and lessons learned

Data Science, Data analytics (IBM, Google), Machine Learning, AI

Official textbook (Data Structures and Algorithms in Java)
- Stack data structure:

LIFO (Last In First Out) data structure
ADT: Abstract data type (interfaces)
push (insert), pop (remove), top (peek), size, isEmpty

Grouping symbol matching problem

- Example#1: Secret Message Application

olleh ereht

h
e
l
l
o

- Example#2: Grouping Symbol matching

Least recently used cache (very famous leetcode problem)

(1 + 2) - [y - 3) --> invalid
([a + b)]--> invalid
([a + b]) ---> valid

Time complexity of the brute force solution: O(n^2)
We use a stack ---> Time complexity: O(n) Space complexity: O(n)

Design it:

a. Static data structure to store the elements: Array
b. Dynamic data structure to store the elements: Singly/Doubly Linked Lists

Object[] array = new Object[4];

["Hi", null, null, null]

capacity: 4
size: 0

Constraint: All the method of the stack must run in O(1)

top = index of the top element
top = -1 = initial value of top index
size = top + 1

Create our own stack from scratch:
1. Create the ADT: interface
2. Create the exception classes: Empty stack and Stack is full
3. Create a class implementing the interface
4. Test the methods out

5. Use the data structure to solve the problem at hand

[( )]

- Everytime we encounter an opening symbol, push it onto the stack
- Everytime we encounter a closing symbol:
a. Stack is empty ==> return false
b. Current symbol and top one are of different nature ==> return false;
c. Else: pop the top element

If we end up with an empty stack ==> valid; otherwise ==> expression is not valid
Time complexity: O(n)