we now have 2 MSEE tutors
Lab1 closed, Lab2 open.
Lab: min ½ max functions.

List of Into ADT:

- createList()
create a new empty list.
- isEmpty()
determine whether a list is empty
- size()
determine length of list.
* add(index, item)
  inserts item into position index.

* remove(index)
  deletes item at position index.

* removeAll()
  deletes all items, putting list into empty state.

* get(index)
  returns item at position index.

It's understood that indices are in range

\[1 \leq \text{index} \leq \text{size()}) \leq \text{MAX\_LENGTH}\]
when an item is inserted using `add()`, the items to its right are shifted to the right, their indices are increased.

Likewise when we delete using `remove()`, everything to the right shifts left, indices decrease.
Ex.

createList() ()

isEmpty(1) (1) true

add(1, 5) (5)

add(2, 3) (5, 3)

add(3, 9) (5, 3, 9)

add(2, 7) (5, 7, 3, 9)

size() (5, 7, 3, 9) 4

get(2) (5, 7, 3, 9) 7

remove(3) (5, 7, 9)

isEmpty() (5, 7, 9) false

removeAll() ()
access function:
an operation that returns information about the state, but does not change it.

manipulation procedures:
operations that alter the state, return nothing.
Implementation using an array

array

ind. 0 1 2

5 3 9 ...

List

1 2 3

max-length - 1

size()

add(2, 7)

5 7 3 9

max-length

1 2 3 4

size()

remove(1)

7 3 9 9

1 2 3

size(1)
Pre-conditions:

Conditions imposed on each operation.

For instance: `add(1, 1)` cannot be called if `size / 1 == MAX_LENGTH`, and `remove(1)` is undefined if `isEmpty(S)` is true.

We establish pre-conditions for ADT operations:

Policy: When pre-cond. are violated, kill program with an error message.
modularity

an ADT is a black box:

A module is part of a program that is separated from the rest of the program by a well defined interface.
services are said to be exported through the interface.

Information hiding means the client cannot see inside the black box.
In Java, an ADT is embodied in a class.

A Java class consists of:

- **fields** - member variables
- **methods** - member functions

We use **private** to implement int. hiding.

Interface consists of **public** (or **package**) methods.