# 4. [] -> add(A) -> [A] @KELSI

Tests:

// Adding front/back

addToFront(X) throws no Exception -> [X, A]

addToRear(X) throws no Exception -> [A, X]

// addAfter

addAfter(X, Z) throws NoSuchElementException

addAfter(C, A) throws no Exception -> [A, C]

addAfter(D, B) throws NoSuchElementException -> [A, D] `// Exception B is not in array`

addAfter(E, B) throws NoSuchElementException -> [E, A] `// Exception B is not in array`

// add(Char)

add(X) throws no Exception -> [A, X]

// add(index)

add(-1, X) throws IndexOutOfBoundsException

add(0, X) throws no Exception -> [X, A]

add(1, X) throws no Exception -> [A, X]

add(2, X) throws IndexOutOfBoundsException

// Removing first/last

removeFirst() throws no Exception -> []

removeLast() throws no Exception -> []

// remove(Char)

remove(X) throws NoSuchElementException

remove(A) throws no Exception -> []

// remove index

remove(-1) throws IndexOutOfBoundsException

remove(0) throws no Exception -> []

remove(1) throws IndexOutOfBoundsException

// set(index)

set(-1, X) throws IndexOutOfBoundsException

set(0, X) throws no Exception -> [X]

set(1, X) throws IndexOutOfBoundsException

// get(index)

get(-1) throws IndexOutOfBoundsException

get(0) throws no Exception -> "A"

get(1) throws IndexOutOfBoundsException

// get(2) throws IndexOutOfBoundsException

// Searching Tests

indexOf(X) returns -1

indexOf("A") returns 0

first() returns "A"

last() returns "A"

contains(X) returns false

contains(A) returns true

// Checking Size and State of List

isEmpty() returns false

size() returns 1

// String Representation of List

toString() returns "[A]"

# 15) [A] -> remove(0) -> [] // KELSI

Tests:

// Adding front/back

addToFront(X) throws NoSuchElementException

addToRear(X) throws NoSuchElementException

// addAfter`- this one could probably just have the X example since no elements are in the array, could just copy Sevigny's example verbatim for all of 15`

addAfter(X, Z) throws NoSuchElementException

addAfter(C, A) throws NoSuchElementException

addAfter(D, B) throws NoSuchElementException

addAfter(E, B) throws NoSuchElementException

// add(char)

add(X) throws NoSuchElementException`// We should be able to add since it adds to the first available slot, since it's empty that would be i=0 -> [X] add(char) should never throw an exception.`

// add(index)

add(-1, X) throws IndexOutOfBoundsException

add(0, X) throws NoSuchElementException`// This one should work as well since index !> size. Index = size = 0`

add(1, X) throws IndexOutOfBoundsException

// Removing first/last

removeFirst() throws NoSuchElementException

removeLast() throws NoSuchElementException

// remove(char)

remove(X) throws NoSuchElementException

remove(A) throws NoSuchElementException

// remove(index)

remove(-1) throws IndexOutOfBoundsException

remove(0) throws IndexOutOfBoundsException

// set(index, char)`// probably could just use the -1 and 0 examples per Sevigny's example`

set(-1, X) throws IndexOutOfBoundsException

set(0, X) throws NoSuchElementException

set(1, D) throws NoSuchElementException

set(2, E) throws NoSuchElementException

// get(index)`// probably could just use the -1 and 0 examples per Sevigny's example`

get(-1) returns IndexOutOfBoundsException

get(0) returns NoSuchElementException

get(1) throws NoSuchElementException

get(2) throws NoSuchElementException

// Searching Test

indexOf(X) returns -1

first() throws NoSuchElementException

last() throws NoSuchElementException

contains(X) returns false

contains(A) returns false`// probs could just use X example based on the example`

// Checking Size and State Test

isEmpty() returns true

size() returns 0

// String Representation of List

toString() returns "[]"

16. [A] -> set(0,B) -> [B] //TODO

17. [A,B] -> addToFront(C) -> [C,A,B] //TODO

18. [A,B] -> addToRear(C) -> [A,B,C]

19. [A,B] -> addAfter(C,A) -> [A,C,B]

20. [A,B] -> addAfter(C,B) -> [A,B,C] @KELSI

Tests:

// Adding front/back

addToFront(X) throws no Exception -> [X,A,B,C]

addToRear(X) throws no Exception -> [A,B,C,X]

// addAfter(char, char)

addAfter(X, Z) throws NoSuchElementException

addAfter(X, A) throws no exception -> [A, X, B, C] `// add working examples`

addAfter(X, B) throws no exception -> [A, B, X, C] `//`

addAfter(X, C) throws no exception -> [A, B, C, X] `//`

// add(char)

add(X) throws no Exception -> [A, B, C, X]

// add(index, char)

add(-1, X) throws IndexOutOfBoundsException

add(0, X) throws no Exception -> [X,A,B,C]

add(1, X) throws no Exception -> [A,X,B,C]

add(2, X) throws no Exception -> [A, B, X, C]

add(3, X) throws no Exception -> [A, B, C, X]

add(4, X) throws IndexOutOfBoundsException

// Remove first/last

removeFirst() throws no Exception -> [B,C]

removeLast() throws no Exception -> [A,B]

// remove(char)

remove(X) throws NoSuchElementException

remove(A) returns A `// add working examples`

remove(B) returns B

remove(C) returns C

remove(-1) throws IndexOutOfBoundsException

remove(0) returns A -> [B, C] `// add return for test`

remove(1) returns B -> [A, C] `// add return for test, changed resultant array to have B removed instead of C, set index to 1 as the test will loop through the array`

remove(2) returns C -> [A, B] `// not out of bounds would return C and remove C from resultant array, set index to 2`

remove(3) throws IndexOutOfBoundsException `// Set index to 3, Add example for index out of bounds > size`

// Extra Tests

addAfter(X, A) throws no Exception -> [A, B, C, X] `// changed added value since C is already in the array. original [A, C, B]`

addAfter(X, B) throws no Exception -> [A, B, X, C] `// changed added element to X for consistancy, added C back in resultant array`

addAfter(X, C) throws no Exception -> [A, B, C, X] `// changed added element to X for consistancy`

addAfter(X, Z) NoSuchElementException `// adding back in failing example`

// Modifying Element Test

set(-1, X) throws IndexOutOfBoundsException

set(0, X) throws no Exception -> [X,B,C]

set(1, X) throws no Exception -> [A, X, C] `// typo fix, changed index to 1 to set B`

set(2, X) throws no Exception -> [A, B, C] `// changed from exception and changed to index 2`

set(3, X) throws IndexOutOfBoundsException `// replace index > size example`

// Retrieving Elements Test

get(-1) throws IndexOutOfBoundsException

get(0) returns "A"

get(1) returns "B"

get(2) returns "C"

get(3) throws IndexOutOfBoundsException

// Searching Test

indexOf(X) returns -1

indexOf(A) returns 0 `// add working examples`

indexOf(B) returns 1 `// add working examples`

indexOf(C) returns 2 `// add working examples`

first() returns "A"

last() returns "C"

// contains

contains(X) returns false

contains(A) returns true `// add working examples`

contains(B) returns true `// add working examples`

contains(C) returns true `// add working examples`

// Checking Size and State Test

isEmpty() returns false

size() returns 3

// String Representation of List

toString returns "[A,B,C]"

27. [A,B] -> remove(A) -> [B] // @KELSI

Tests:

// Adding front/back

addToFront(X) throws no Exception -> [X, B]

addToRear(X) throws no Exception -> [B, X]

// addAfter(char, char)

addAfter(X, Z) throws NoSuchElementException

addAfter(X, B) throws no Exception`// add working example`

add(X) throws no Exception -> [B, X]

add(-1, X) throws IndexOutOfBoundsException

add(0, X) throws no Exception -> [X, B]

add(1, X) throws no Exception -> [B, X]

add(2, X) throws IndexOutOfBoundsException

// Removing first/last

removeFirst() throws no Exception -> []

removeLast() throws no Exception -> []

// remove(char)

remove(X) throws NoSuchElementException

remove(B) returns B`// add working example`

// remove(index)

remove(-1) throws IndexOutOfBoundsException

remove(0) returns B -> []` // remove returns the removed element`

remove(1) throws IndexOutOfBoundsException

// Modifying Elements Test

set(-1, X) throws IndexOutOfBoundsException

set(0, X) throws no Exception -> [X]

set(1, X) throws IndexOutOfBoundsException

// Retrieving Elements Test

get(-1) throws IndexOutOfBoundsException

get(0) returns "B"

get(1) throws IndexOutOfBoundsException

// Searching Test

indexOf(X) returns -1

indexOf(B) returns 1`// add working example`

first() returns "B"

last() returns "B"

contains(X) returns false

contains(B) returns True`// add working example`

// Checking Size and State Tests

isEmpty() returns false

size() returns 1

// String Representation of List

toString() returns "[B]"

30. [A,B] -> remove(1) -> [A] @KELSI

Tests:

// Adding front/back

addToFront(X) throws no Exception -> [X,A]

addToRear(X) throws no Exception -> [A,X]

// addAfter(char, char)

addAfter(X, Z) throws NoSuchElementException

addAfter(X, A) throws no exception`// add working example`

//add(char)

add(X) throws no Exception -> [A,X]

// add(index)

add(-1, X) throws IndexOutOfBoundsException

add(0, X) throws no Exception -> [X]

add(1, X) throws no Exception -> [A,X]

add(2, X) throws IndexOutOfBoundsException

// Removing first/last

removeFirst() throws no Exception -> []

removeLast() throws no Exception -> []

// remove(char)

remove(X) throws NoSuchElementException

remove(A) returns A`// add working example`

// remove(index)

remove(-1) throws IndexOutOfBoundsException

remove(0) returns A []`// remove returns removed element`

remove(1) throws IndexOutOfBoundsException

// Modifying Element Test

set(-1, X) throws IndexOutOfBoundsException

set(0, X) throws no Exception -> [X]

set(1, X) throws IndexOutOfBoundsException

// Retrieving Element Test

get(-1) returns IndexOutOfBoundsException

get(0) returns "A"

get(1) throws IndexOutOfBoundsException

// Searching Test

indexOf(X) returns -1

indexOf(A) returns 0 // add working test

first() returns "A"

last() returns "A"

contains(X) returns false

contains(A) returns true`// add working test`

// Checking Size and State Test

isEmpty() returns false

size() returns 1

// String Representation of List

toString() returns "[A]"