Create eclipse project 2190221\_Q1\_{yourID}\_{FirstName}, for example, **2190221\_Q1\_6532782021\_Adam**

**Write code for class Deck** that represents a deck of cards in a card game while playing (Each card is just a non-negative integer value). **You must create class Deck from scratch**.

* Choosing existing data structures is important (choose by considering performance).
  + The chosen data structures must ONLY come from the folders below (you can copy it to your project):

A screenshot of a computer screen

Description automatically generated

**If you use other data structures outside these folders,** **you get 0 points.**

**If you create new array or arraylist, you get 0 points.**

* + You can **mix files** from the above folders.
  + **JUnit** test file is **in folder Q1**.
* You MUST NOT modify all given data structures files. **You get 0 mark if you do modify it/them**.
* You can write new class(es) that extends from given class(es).

Class Deck stores a deck of cards. Its operations are as follows (All methods MUST NOT throw Exception). Use try/catch to surround the area with potential exception.

* public int draw():
  + If there is no card to remove, return -1.
  + remove a card from the top of the deck. Return the value of that card.
* public int removeNth(int n):
  + remove the nth card (and return its value). The top card is the 0th card. Assume n is always non-negative.
  + If the nth card does not exist, return -1 and do nothing.
* public void putBottom(int n):
  + Put card with value n at the bottom of the deck. This is used to create a deck in the test cases.
* public void reverseTopN(int n):
  + reverse the order of the top n cards (position 0 to position n-1, inclusive). Assume n is positive.
  + If n is too large, just reverse the entire deck. If the deck is empty, do nothing.
  + for example, if the cards are originally:



reverseTopN(3) will give us:

|  |
| --- |
| 3 |
| 1 |
| 6 |
| 9 |
| 4 |

|  |
| --- |
| 6 |
| 1 |
| 3 |
| 9 |
| 4 |

**Scoring Criteria:**

The total score is 25 (will be scaled to 10).

Choosing efficient data structure 5 marks

Minimizing number of iterations in your code 3 marks

Run the given JUnit files (If you do not write your code, you will not get any marks):

* testDraw 1 mark
* testPutBottom 1 mark
* testRemoveNthFirst 1 mark
* testRemoveNthOut 1 mark
* testRemoveNthLast 1 mark
* testRemoveNthGeneric 4 marks
* testReverseEmptyDeck 1 mark
* testReverseEntireDeck 3 marks
* testReverseGeneric 4 marks