- Don't forget to set your Eclipse workspace and working set.
- You must submit the JAR file, exported (with source code), from your Eclipse project.
- You must check your JAR file to make sure all the source files (.java files) are present. It can be opened with file compression programs such as 7-zip or Winrar.
- Failure to export properly will result in your work not getting marked.

1) To submit:
    - Export your project to a JAR file, with source code.
    - Name your JAR file ID_Week05_Q1.jar. For example, 6623110021_Week05_Q1.jar
    - Submit the JAR file on MyCourseville.

Instructions:

Create Eclipse project of your choice.

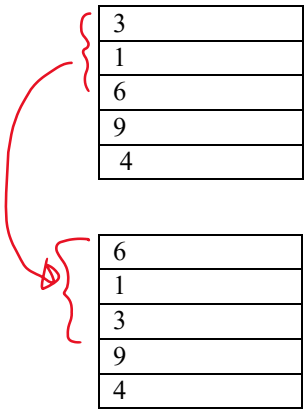
Write code for class Deck that represents a deck of cards in a card game while playing (Each card is just a non-negative integer value). **You must create class Deck from scratch**.

- **DeQLinkedList is a class extened from QueueLinkedList. QueueLinkedList contains a circular linked list inside.**
- **DeQLinkedList provides methods for inserting/removing data at the front/back of the linked list. Please look at the source code for detail.**
- Your task is to **use DeQLinkedList to implement Deck**.
    - **JUnit** test file is **already available.**
- **If you create new array or arraylist, you get 0 mark.**
- You MUST NOT modify all given data structures files. **You get 0 mark if you do modify it/them**.
- **If you do not write Deck using DeQLinkedList, you get 0 mark.**
- **Each Unnecessary loop will get 1 mark deducted.**
- 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. You must try-catch properly when calling methods such as next() or previous()).

- 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 0^{th} 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:



Scoring Criteria:

The total score is 17 (will be scaled to equal to other homework).

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