Notice

- We cannot accept late submissions for **any excuse**.
- Please visit the AR website on academic integrity.
- For each question, write the following information as a comment preceding the method:
  1) all the Internet resources you've used for solving the problem (even though you didn't explicitly copy from them);
  2) all the students with whom you have discussed this question; and
  3) running time of your method, with the smallest possible function.[1]
- Make your algorithms as efficient as possible, in terms of both time and space, with priority on time. Your scores depend on their efficiency.
- No points shall be awarded if your submission does not compile.[2]
- Submission procedure (each deviation will result in a deduction of 10 points):
  1) Name the .java file as `<class_name>_<your_id>_<your_name>.java`, e.g., `GradeBook_12345678d_TangTszkei.java` if your name is Tang Tszkei and your ID is 12345678d. (You may find Alt-Shift-R helpful if you're using Eclipse.)
  2) Put all your files into a folder with name `A1_<your_id>_<your_name>`.
  3) Create a .zip or .jar file to contain this folder, similar as the distributed file. **Warning: only zip and jar are accepted**.

[1] A frequent question is "Whether I can use methods from Java library?" Most library methods are complicated and difficult to analyze. If you use them, you have to count their steps, which is almost impossible from my experience.

[2] If you've problems, this and this web pages may help, and you're welcome to seek help on the discussion forum (DON'T SHARE YOUR CODE).

1. (40 points) Grader: Pan Chao (chao.pan@connect.polyu.hk)

   We are playing special cards, which have the usual four suits (Spades, Hearts, Clubs, and Diamonds) but the ranks can be any positive integer. You have been delivered a hand of n cards, and you have sorted them in the suit-first order: Spades, Hearts, Clubs, and Diamonds, each suit in non-increasing order.

   Write an algorithm to reorder your hand of cards into rank-first order: for cards of the same rank, you follow the order of Spade, Heart, Club, and then Diamond. An example can be found in `CardGame.java`.

   In your algorithm, you are **not allowed** to create new cards by calling `new Card()`.

   ```
   void reorder(Card[] hand) {}
   ```

2. (60 points) Grader: Zhao Xiangyu (xiang-yu.zhao@connect.polyu.hk)

   In Lab 5 you learned how to simulate a linked list with an array. This question is about simulating a doubly linked list with an array.

			2			5			8			11			14			17
(1)	0	0	0		5	2		8	5		11	8		14	11		0	2
(2)	5	2	5	75	0	0	99	2	0		11	8		14	11		0	8
(3)	5	11	5	75	8	0	99	2	2	85	11	8	38	0	0		0	14
(4)	2	11	0	75	8	~~0~~	~~99~~	~~14~~	2	85	11	8	38	0	5		0	5
(5)	2	5	0	75	8	11	49	0	2	85	11	8	38	5	0		0	14

   In the example, we use an array `arr` of length 18 and it can store 6 nodes. The indices in `arr[0]` and `arr[1]` are the *head* and *tail* respectively, and they are both 0 when the list is empty. We need actually another list to maintain the unused nodes, but this time we don't need to keep track both its head and tail. We use the last slot, i.e., `arr[n-1]`, as the head of the list for unused nodes. Illustrated above are the status of the array (blue for numbers and maroon for indices)

   (1) after the initialization,

   (2) after the execution of `insertFirst(75); insertFirst(99);` (note their order),

   (3) after the execution of `insertLast(85); insertLast(38);` (note their order),

   (4) after the execution of `deleteFirst();` (note that the second node, `arr[5]`–`arr[7]`, is recycled and put back to the head of the unused list), and

   (5) after the execution of `insertLast(49);`.

   Implement all methods in `DListOnArray.java`. All of them run in $O(1)$ time.

   (Bonus question of 4 points. Write an $O(n)$-time algorithm to reverse the list. For example, if it is called after step (5), then the list becomes 49, 38, 85, 75.)