GIT Department of Computer Engineering
CSE 222/505 - Spring 2020
Homework 4_Q2/Q3

BARAN HASAN BOZDUMAN
171044036

Problem Approach for Q2

Firstly they wanted from us to implement a deque, deque is like stack and queue at the same time so we can add and remove both side of list. And there are many methods which are doing same operation such as there is a three add first method they do the same job but They differs when the list has restrictions such as size etc they gives different reaction. For instance if the capacity is full some of them throws exception some of them returns false so according to your necessity you use different methods.Since there is no restrictions the given problem almost all adds just adds the value does not return boolean or it does not throws exception. But the remove method does if there is no such an element it throws exception or some of them just returns false I implemented it

As a second part the given problem we do not assign the nodes just null an leave it garbage collector we are doing our job so we keep the removed nodes in another linked list and after a removed operation if user wants to add new value to deque we take a node from the removed nodes list and link it to deque and I think that I keep remove double linkedlist too and also I keep removed nodes tail so if user demand a new node I look the my removed nodes list if its availabla I take the last of it and link it to deque list so we dont need to create node each time.

Q2 -Test Case Table

Test number	Number of Output Image	Step Details	Expected Result	Actual Result	Pass/Fail/Not Executed
1	1	Add elements with addFirst() method	Added 1, 2, 3,4,5	Added last successfully	PASS
2	2	Add elements with add() method	Adding 6, 7, 8,9,10	Added last succesfully	PASS
3	3	Add elements with addLast() method	Adding 11,9,3,12,13.14. 15,10001	Added last succesfully	PASS
4	4	Add elements with offerFirst() Method and print boolean values	Adding 0, -1 , -2, -3, -4,	Added begin of deque succesfully	PASS
5	5	Call removeFirst method three times	Removing first three elements	Removed first three elements successfully	PASS
6	5	Call pollLast method once	Removing last element	Removed last element successfully	PASS
7	6	Call poll method twice	Removing first two elements	Removed first two successfully	PASS
8	6	Call pollFirst method once	Removing first element	Removed first one successfully	PASS
9	7	Call remove first occurance individual element which is in list print boolean value	Removing the element	Find and removes the elements	PASS
10	7	Call remove first occurance for the two same element	Removing first occurance	Removed first occurance	PASS

		which is in list print boolean value		successfully	
11	7	Call remove first occurance element which is not in the list print boolean value	Returning false	Returned false	PASS
12	8	Call remove last occurance individual element which is in list print boolean value	Removing the element	Find and removes the elements	PASS
13	8	Call remove last occurance for the two same element which is in list print boolean value	Removing last occurance	Removed last occurance successfully	PASS
14	8	Call remove last occurance element which is not in the list print boolean value	Returning false	Returned false	PASS
15	9	Call the offer method	Insert end of the list	added succesfully	PASS
16	9	Call the offerlast method	Insert end of the list	Added successfully	PASS
17	9	Calling add method after removed something	It add rhe last removed element and changing data and insert to deque	It took node from removed elements and added into deque	PASS
18	10	Call Offer first method twice	It inserts two element	It inserted successfully	PASS

			beginning of the deque		
19	10	When adding element take it from removed node	Call add element and check the removed elements are decreases or not	It decreases	PASS
20	11	Creating an iterator and print list with next and hasNext	Prints the entire list	It printed the entire list	PASS
21	12	Creating a descending iterator and print list with next and hasNext	Prints entire list in reverse order	It printed the entire list in reverse order	PASS
22	13	Create an ascending iterator and call next operator four times and remove it	It removes third index	It removed third index	PASS
23	14	Create an descending iterator and call next operator five times and remove it	It removes forth index in backward	It removed forth index in backward	PASS
24	15	Contains method with value in the list	Returns true	Returned true	PASS
25	15	Contains method with value in the removed list	Returns false	Returned false	PASS
26	15	Contains method with value which	Returns false	Returned false	PASS

		is not in any list			
27	16	Call peek method	Returns first value	Returned first value	PASS
28	16	Call peekfirst method	Returns first value	Returned first value	PASS
29	16	Call peeklast method	Returns last value	Returnedlast value	PASS
30	16	Call getFirst	Returns first vallue	Returned first vallue	PASS
31	16	Call getlast	Returns last value	Returned last value	PASS
32	16	Check the deque is empty method	false	false	PASS
33	17	Take it in a try catc and remove even tere is no element	Throws nllptr exception	Throwed nllptr exception	PASS
34	17	Call is empty method	true	true	PASS
35	17	After reove all element Take it in a try catch block and look peek method			PASS
36	17	After reove all element Take it in a try catch block and look peek and peekfirst method	Throws nosuch element exception	Throwed nosuch element exception	PASS
37	17	After reove all element Take it in a try catch block and call peeklast method	Throws nosuch element exception	Throwed nosuch element exception	PASS
38	17	After reove all element	Throws nosuch	Throwed nosuch	PASS

		Take it in a try catch block and call get first method	element exception	element exception	
39	17	After reove all element Take it in a try catch block and call get last method	Throws nosuch element exception	Throwed nosuch element exception	PASS

Problem Approach for Q3

For each questions we need a base case to stop the functions to return back the results. It works for a specific problem or value, after that we can show it works for bigger problems too like proof of inductions, if it works for n=0 or n=1 and we assume that it works for n, Also it must wor n+1 too. Some methods may look complicated but I have to do that to make my all methods recursive.

Q3.1

-Identify the base case

For the first question our base case if the index end of the string it returns -1(because it does not have more white space) and it starts to stop the recursive methods wich are in stack and it prints the word when it goes the backward because we print them after return new method calling.

-Define the smaller problem

It works for a word it prints only it.

-Explain how to combine solutions

After you make work it for a smaller method such as two word it also works for other.

Q3.2

-Identify the base case

For the check is elfish word I used a helper method which goes end of the string(which is the base case of it end of the string) and check has it the giveen letter and I make it for eack letter and returns the and operation of them.

-Define the smaller problem

It works for a letter

-Explain how to combine solutions

So I can use it to check each letter seperately and combine them with add operator.

Q3.3

-Identify the base case

To sort an array I used two recursive helper one of them is takes the next number in array and send it to another method and itchecks if there is a smaller than that number if it has it returns the index of it and swap them and goes the next element if it is in the last index it stops method.

-Define the smaller problem

It can works for an array which has two element if the first is bigger then second it swaps.

-Explain how to combine solutions

It checks each element ifor current index and look for the smaller number. So if it works two size array it works for n too.

Q3.4/5

-Identify the base case

They uses same helper methods which is base case end of the string but for evaluating perfix in prefix wrapper I send the reverse of string into he helper methods so it calculate it as postfix and I send the ispostfix parameter as boolean so when it makes the operation according to given variable for the dividing and substraction.

-Define the smaller problem

It works for one operation such as +54.

-Explain how to combine solutions

If it does the operations accurately for small problems it works for bigger problems too.

Q3.6

-Identify the base case

I uses for sub helper methods for print and one main helper method It base case if the size is smaller than or equal to 0 it stops and I decrease it for each print operation note that My all helper methods are recursive methods.

-Define the smaller problem

It prints firstly top of the matrix and then right column bottom and left column and each time I increase row length by one to keep spirallpattern so for 2x2 matrix it first print 12 then 3 then4 and since the size is zero it stops.

-Explain how to combine solutions

Since we can apply it for smaller problems we can apply it for bigger mxn marices too.

Q3 -Test Case Table

Test number	Number of Output Image	Step Details	Expected Result	Actual Result	Pass/Fail/Not Executed
1	1	Try it 3 word	It reverse the word order	It works	PASS
2	1	Try it with loots of spaces asymmetric	It calculate space too	It print reverse order whitespacces include	PASS
3	1	Try with nuber of strings to easy see	Prints reverse order the numbers	It printed numbers in reverse order	PASS
4	2	Try it words have elf	Returns true	Returned true	PASS
5	2	Try with lower and upper case	Returns true	Returned true	PASS
6	2	Try with lack of one letter	Returns false	Returns false	PASS
7	3	Try with different number of unsorted arrays and prin the array	It prints in order	It printed in order	PASS
8	3	Try with negative numbers	It prints in order	It printed in order	PASS
9	3	Try with which has same number	It prints in order	It prints in order	PASS
10	4	Try it with	print actual	They match	PASS

		different prefixes	result and method result		
11	5	Try it with different postfixes	print actual result and method result	They match	PASS
12	6	Try it with 2n+1X2n+1	7x5 and print last number to easy read	It works	PASS
13	6	Try it with 2n+1X2n	5x4 and print last number to easy read	It works	PASS
14	6	Try it with 2nX2n	4x4 and print last number to easy read	It works	PASS
15	6	Try it with 2nX2n+1	4x5 and print last number to easy read	It works	PASS
16	6	Try it with 2nX2n+1	6x3 and print last number to easy read	It works	PASS

PS:

- The java doc files in the directory of Q1/2.
- If you want to check the test case table results you can check the relatived directory in report directory.
- The Class diagrams are also in report directory as .png files.