CS213: Data Structures and Algorithms **Deadline: 9:00 PM, Feb-20 2021**

Task #3

Instructor: Prof. Sharat Chandran

1 What grows up must shrink down!

We have provided a partial implementation for Dynamic Array in python. The Array doubles in size whenever an append request overflows its current capacity. However, the array does not shrink its capacity on receiving a remove request. Your task is to implement this array shrinking functionality.

One requirement is that you should satisfy an **invariant** that the capacity of the array at any time should be a **power of 2**.

As part of this question, you are expected to do the following three steps:

1. Download the python code from this link and paste it in your Hackerrank console.
2. Add code to the `remove` function in the `DynamicArray` class to call *shrink_array* function.
3. Add code in *shrink_array* function to reduce the capacity of the array.

Note that all other parts of the provided code should be **unaltered**. Understand the code completely and reuse the existing functions as much as you can. Redundant implementation will attract penalty as appropriate.

Reflection Essay:

Explain what you set up as your goals for the shrinking in your reflection essay and whether you achieved your goals satisfactorily.

2 Deleting a node in a Singly Linked List

Given a singly linked list (i.e, you have been provided access to the head node) and a pointer to any node in the list, delete the node at the given pointer and return the remaining list. It is guaranteed that the node to be deleted will not be the last node.

For implementing this, you have been given a code template here and you must use that template only. You have to complete the function `deleteNode` and not modify anything else.

Example: Input list

$10 \longrightarrow 20 \longrightarrow 30 \longrightarrow 40 \longrightarrow 50$
You have a pointer to the head element 10, and pointer to the 4th element 40 that is to be deleted.

The output list is

$10 \longrightarrow 20 \longrightarrow 30 \longrightarrow 50$
Reflection Essay:

You should give a brief explanation of your solution and understanding of this problem in the reflection essay. For obtaining full marks on hackerrank, your solution must be as fast as it can ever be.

3 Span

Given an an array X, the span S[i] of X[i] is the maximum number of consecutive elements X[j] immediately preceding X[i] and such that X[j] <= X[i]

Write a linear time program using a stack that computes the span

Example:

Input: X = 6, 3, 4, 5, 2

Output: S = 1, 1, 2, 3, 1

Explnation: The number of elements less than or equal to 4 is 2 if we consider only items to the left of 4 (and 4 itself).

Note: Your program must use a Stack ADT for credit and use it to solve the problem.

Reflection Essay:

Justify your run time performance.

4 Too many Queries

This question is marked as a starred question.

You have an array a of N integers $a_1, a_2, \ldots, a_N$. You are next given Q queries, each of which is one of the following types:

- `I i x` - Insert the number x at index i, thus increasing the length of array by 1.
- `D i` - Delete the number at index i, thus decreasing the length of array by 1.
- `R 1` - Rotate the array clockwise by 1 unit (a_i shifts to index $i+1$ for $1 \leq i \leq n-1$ and a_n shifts to index 1, where n is the current length of the array).
- `R -1` - Rotate the array counterclockwise by 1 unit (a_i shifts to index $i-1$ for $2 \leq i \leq n$ and a_1 shifts to index n, where n is the current length of the array).
- `P i` - Print the number at index i.

Constraints:
$1 \leq N, Q \leq 10^5$
$0 \leq a_1, a_2, \ldots, a_N, x \leq 10^9$
$1 \leq i \leq$ current length of array
Your task is to execute the Q queries and output the print queries to the console. All these queries should be performed in an efficient manner.

NOTE: If you use in-built deque data structure, only 20% credit will be given. Try to make use of doubly linked lists.

Reflection Essay:

Did you encounter any difficulty in the hacker rank portal? Is that because of your implementation? Think of a super fast implementation.

5 Amortized Analysis (Written)

Consider a sequence of stack operations that guarantees that the stack will not exceed n entries. After every n operations, a backup of the stack is made. What is the amortized time complexity of p stack operations?

Note: Give reference to the technique mentioned in the video `amortized.mkv`

6 Secret ADT (Written)

This question is marked as a starred question.

You are given a function, f : X $\rightarrow$ Y, where X = {0,1...N} and Y = {1,2...,N}

This function is onto, i.e., for every y $\in$ Y there is a x $\in$ X such that f(x) = y.

Clearly there will exist exactly one element y $\in$ Y such that distinct x_1, x_2 $\in$ X satisfy f(x_1) = f(x_2) = y.

The goal is to find the elements x1,x2 and y defined above. You are given N.

Problem Constraint: Someone has implemented two data structures that you are **required** to use to solve this problem.

1. A secret ADT. This ADT implements a member function f. i.e., it returns f(x) when given x.
2. Two instances of a truncated linked list ADT. This linked list has a capacity of size 2.

Note: We can solve the problem in $O(N)$ space easily if we use an additional array. So the goal is to use only $O(1)$ space and the data structures given above to solve this problem. We do not know or care how much time or space these ADT provided take i.e. we do not count these in time or space.

Hint: The intent of the linked list is to store x and f(x) in x.next

No credit will be given to solutions which use more space. If you are not clear what O(1) space means, please clarify with your TA.