CSE 4304
Data Structures Lab
Lab 9 - Segment Tree
Winter, 2020

Task 1:

Implement the Initialization, Query and update operations of a Segment Tree and test for different examples as demonstrated in the lecture videos.

Make sure that you understand the intermediate steps of the recursive algorithm. Moreover, you should also able to draw the trees from a set of values.

Task 2:

Given an array with **N** elements, indexed from $1 \ldots N$. Now you will be given some queries in the form $\boldsymbol{I} \ldots \boldsymbol{J}$, your task is to find the minimum value from index $\boldsymbol{i}\ to\ \boldsymbol{j}$.

Input:
Each of the test cases will have two values N, Q in first line denoting the total number of elements(N) and the total number of queries(Q).
Next line will take N numbers.
Each of the following Q lines will have two values indicating the values of $\boldsymbol{i}$ & $\boldsymbol{j}$.

Output:
For each of the Queries, you have to print the minimum value from index $\boldsymbol{i}$ & $\boldsymbol{j}$.

Sample Input	**Sample Output**
5 3 78 1 22 12 3 1 2 3 5 4 4	1 3 12
1 1 10 1 1	10
6 6 20 50 10 40 90 30 1 6 3 3 5 5 5 6 4 6 3 6	10 10 90 30 30 10

Note: Each of the queries should be solved in $O(logn)$ time.

Task 3:

Robin Hood likes to loot rich people since he helps the poor people with his money. Instead of keeping all the money together, he does another trick. He keeps $\boldsymbol{n}$ sacks where he keeps the money. The sacks are numbered as $1 \ldots n$.

Now each time he can do one of the three tasks.

1. Give all the money of the i^{th} sack to the poor, leaving the sack empty.
2. Add a new component (given in input) into the j^{th} sack.
3. Find the total amount of money from i^{th} sack to j^{th} sack.

Since he is not a programmer, he seeks your help.

Input:
Each of the test cases will have two values N, Q in first line denoting the total number of elements (N) and the total number of queries(Q).
Next line will take N numbers.
Each of the following Q lines which can have values as:

i) $1\ i$ (For this case, give all money from ith sack to the poor.)
ii) $2\ i\ v$ (Add v amount of money to the i^{th} sack.
iii) $3\ i\ j$ (Find the total amount of money from i^{th} to j^{th} sack.)

Output:
If the query is type-1, print the amount of money that will be given to the poor. If the query is type-3, print the total amount from $i\ to\ j$.

Sample Input	**Sample Output**
5 6 3 2 1 4 5 1 4 2 3 4 3 0 3 1 2 3 0 4 1 1	5 14 1 13 2

Note: Each of the queries should be solved in $O(logn)$ time.