**Assignment 8**

**Q1:- A Barua number is a number which consists of only zeroes and ones and has only one 1.**

**Barua number will start with 1. Given numbers, find out the multiplication of the numbers.**

**Note: The input may contain one decimal number and all other Barua numbers. (Assume**

**that each number is very large and total number of values give is also very large)**

**Input 1: 100 10 12 1000**

**Output 1: 12000000**

**Input 2: 100 121 1000000000000000**

**Output 2: 12100000000000000000**

**Input 3: 10 100 1000**

**Output 3: 1000000**

**Answer :--**

**Q 2 :-- Implement push, pop and find the minimum element in a stack in O(1) time complexity.**

**Answer:-** A variable **minEle** that stores the current minimum element in the stack. Now the interesting part is, how to handle the case when minimum element is removed. To handle this, we push “2x – minEle” into the stack instead of x so that previous minimum element can be retrieved using current minEle and its value stored in stack.

**Push(x)** : Inserts x at the top of stack.

* If stack is empty, insert x into the stack and make minEle equal to x.
* If stack is not empty, compare x with minEle. Two cases arise:
  + If x is greater than or equal to minEle, simply insert x.
  + If x is less than minEle, insert (2\*x – minEle) into the stack and make minEle equal to x. For example, let previous minEle was 3. Now we want to insert 2. We update minEle as 2 and insert 2\*2 – 3 = 1 into the stack.

**Pop() :**Removes an element from top of stack.

* Remove element from top. Let the removed element be y. Two cases arise:
  + If y is greater than or equal to minEle, the minimum element in the stack is still minEle.
  + If y is less than minEle, the minimum element now becomes (2\*minEle – y), so update (minEle = 2\*minEle – y). This is where we retrieve previous minimum from current minimum and its value in stack. For example, let the element to be removed be 1 and minEle be 2. We remove 1 and update minEle as 2\*2 – 1 = 3.