# Data Structures and Algorithm | Jan 2021

# Assignment 6 | 23rd January 2021

## Question 1

A Barua number is a number that consists of only zeroes and ones and has only one 1. Barua's 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 the very large and the 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:-

```
a = [int(i) for i in input('Enter values: ').split()]
n = len(a)
result = 1
for i in range(n):
    result = result*a[i]
print(result)
```

## Question 2

Implement push, pop and find the minimum element in a stack in O(1) time complexity.

## Answer:-

## class Node:

```
        def __init__(self, value):
                self.value = value
                self.next = None
        def __str__(self):
                return "Node({})".format(self.value)
        __repr__ = __str__
class Stack:
        def __init__(self):
                self.top = None
                self.count = 0
                self.minimum = None
        def __str__(self):
                temp = self.top
                out = []
                while temp:
                        out.append(str(temp.value))
                        temp = temp.next
                out = '\n'.join(out)
                return ('Top {} \n\nStack :\n{}'.format(self.top,out))
        __repr__=__str__
        def getMin(self):
                if self.top is None:
```

```
                return "Stack is empty"
            else:
                print("Minimum Element in the stack is: {}"
.format(self.minimum))
        def isEmpty(self):
            if self.top == None:
                return True
            else:
                return False
        def __len__(self):
            self.count = 0
            tempNode = self.top
            while tempNode:
                tempNode = tempNode.next
                self.count+=1
            return self.count
        def peek(self):
            if self.top is None:
                print ("Stack is empty")
            else:
                if self.top.value < self.minimum:
                    print("Top Most Element is: {}" .format(self.minimum))
```

```
            else:
                print("Top Most Element is: {}" .format(self.top.value))
    def push(self,value):
        if self.top is None:
            self.top = Node(value)
            self.minimum = value
        elif value < self.minimum:
            temp = (2 * value) - self.minimum
            new_node = Node(temp)
            new_node.next = self.top
            self.top = new_node
            self.minimum = value
        else:
            new_node = Node(value)
            new_node.next = self.top
            self.top = new_node
        print("Number Inserted: {}" .format(value))
    def pop(self):
        if self.top is None:
            print( "Stack is empty")
        else:
            removedNode = self.top.value
```

```
                self.top = self.top.next
                if removedNode < self.minimum:
                        print ("Top Most Element Removed :{} "
.format(self.minimum))
                        self.minimum = ( ( 2 * self.minimum ) - removedNode )
                else:
                        print ("Top Most Element Removed : {}"
.format(removedNode))
stack = Stack()
stack.push(3)
stack.push(5)
stack.getMin()
stack.push(2)
stack.push(1)
stack.getMin()
stack.pop()
stack.getMin()
stack.pop()
stack.peek()
```