**Assignment 8**

**Question 1**

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

#include <bits/stdc++.h>

using namespace std;

// function to count

// all permutations

void countPermutations(int N, int B)

{

// count of

// all permutations

int x = pow(B, N);

// count of permutations

// with leading zeros

int y = pow(B, N - 1);

// Return the permutations

// without leading zeros

cout << x - y << "\n";

}

// Driver code

int main()

{

int N = 6;

int B = 4;

countPermutations(N, B);

return 0;

}

**Question 2**

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

#include <bits/stdc++.h>

using namespace std;

// A user defined stack that supports getMin() in

// addition to push() and pop()

struct MyStack

{

stack<int> s;

int minEle;

// Prints minimum element of MyStack

void getMin()

{

if (s.empty())

cout << "Stack is empty\n";

// variable minEle stores the minimum element

// in the stack.

else

cout <<"Minimum Element in the stack is: "

<< minEle << "\n";

}

// Prints top element of MyStack

void peek()

{

if (s.empty())

{

cout << "Stack is empty ";

return;

}

int t = s.top(); // Top element.

cout << "Top Most Element is: ";

// If t < minEle means minEle stores

// value of t.

(t < minEle)? cout << minEle: cout << t;

}

// Remove the top element from MyStack

void pop()

{

if (s.empty())

{

cout << "Stack is empty\n";

return;

}

cout << "Top Most Element Removed: ";

int t = s.top();

s.pop();

// Minimum will change as the minimum element

// of the stack is being removed.

if (t < minEle)

{

cout << minEle << "\n";

minEle = 2\*minEle - t;

}

else

cout << t << "\n";

}

// Removes top element from MyStack

void push(int x)

{

// Insert new number into the stack

if (s.empty())

{

minEle = x;

s.push(x);

cout << "Number Inserted: " << x << "\n";

return;

}

// If new number is less than minEle

if (x < minEle)

{

s.push(2\*x - minEle);

minEle = x;

}

else

s.push(x);

cout << "Number Inserted: " << x << "\n";

}

};

// Driver Code

int main()

{

MyStack s;

s.push(3);

s.push(5);

s.getMin();

s.push(2);

s.push(1);

s.getMin();

s.pop();

s.getMin();

s.pop();

s.peek();

return 0;

}