DAY-8 ASSIGNMENT | 2nd January, 2021

1.Problem Statement:
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)

Program in C:

```
#include <iostream>
#include <math.h>
using namespace std;
void findNumberOfDigits(long n, int base)
{
	// Calculating log using base
	// changing property and then
	// taking it floor and then
	// adding 1.
	int dig = (int)(floor( log(n) /
                                        log(base)) + 1);

	cout << "The Number of digits of "
		<< "Number " << n << " in base "
		<< base << " is " << dig;
}

int main()
{
	// taking inputs
	long n = 1446;
	int base = 7;
	findNumberOfDigits(n, base);
	return 0;
}
```

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

Program in C:

```
#include<stdio.h>
int main_stack[100] , supporting_stack[100];

int push(int element , int *top , int *stack)
{
	*top = *top + 1;
	stack[*top] = element;
```

```
}
int pop(int *stack , int *top)
{
	int element;
	if(*top > -1)
	{
	element = stack[*top];
	*top = *top - 1;
	return element;
	}
	else
	{
		printf("\n== STACK EMPTY == \n");
		return -99999;    // means nothing is popped
	}
}
int main()
{
	int choice , element , top_main=-1 , top_support=-1 ,i ,supp_stack_pop_element,
popped_element;
	printf("Enter the operation : \n 1.Push \n 2.Pop \n 3.check minimum \n 4.STOP");
	scanf("%d",&choice);
	while(choice != 5)
	{
		if (choice == 1)
		{
			printf("\nEnter the number to be pushed: ");
			scanf("%d",&element);
			push(element , &top_main , main_stack);
			if (top_support >= 0 && element < supporting_stack[top_support])
			{
				push(element , &top_support , supporting_stack);
			}
			else if (top_support == -1)
			{
				push(element , &top_support , supporting_stack);
			}
		}
		else if (choice == 2)
		{
			popped_element = pop(main_stack , &top_main);
			if (popped_element != -99999)
				printf("\n Popped element = %d",popped_element);
			if (popped_element != -99999)
			{
				if (popped_element == supporting_stack[top_support])
				{
```

```
                                        supp_stack_pop_element = pop(supporting_stack ,
&top_support);
                                }
                        }
                }
                else if( choice == 3)
                {
                        if (top_support > -1)
                                printf("\nMinimum element in the stack = %d \n\n" ,
supporting_stack[top_support] );
                        else
                                printf("\n ==== Stack Empty ======");
                }
                else if (choice == 4)
                {
                        if (top_main > -1)
                                {
                                printf("\n === Main Stack === \n ");
                                for (i=top_main;i>=0;i--)
                                        {
                                                printf("\n%d",main_stack[i]);
                                        }
                                }
                        else
                                printf(" \n ==== STACK EMPTY ===  \n ");
                        printf("\n top_support = %d", top_support);
                        for (i=top_support;i>=0;i--)
                                {
                                        printf("\n%d",supporting_stack[i]);
                                }
                }
        printf("\nEnter the operation : 1.Push \n 2.Pop \n 3.check minimum \n 4.See Full
Stack \n 5.STOP");
        scanf("%d",&choice);
        }
        return 0;
}
```

Output:

```
Enter the operation :
 1.Push
 2.Pop
 3.check minimum
 4.STOP1
 Enter the number to be pushed: 25
 Enter the operation : 1.Push
 2.Pop
 3.check minimum
 4.See Full Stack
 5.STOP1

Enter the number to be pushed: 74
Enter the operation : 1.Push
 2.Pop
 3.check minimum
 4.See Full Stack
 5.STOP3
 Minimum element in the stack = 25


Enter the operation : 1.Push
 2.Pop
 3.check minimum
 4.See Full Stack
 5.STOP5
```