**Data Structure and Algorithms**

**(HackerEarth codemonk solution) 2022**

**Jagruti Suthar**

**AIMDek Trainee**

**Arrays & Strings**

**Q-1 Monk and Rotation**

**URL:** [**https://www.hackerearth.com/problem/algorithm/monk-and-rotation-3-bcf1aefe/**](https://www.hackerearth.com/problem/algorithm/monk-and-rotation-3-bcf1aefe/)

**Solution:**

import java.util.\*;

public class test {

public static void main (String[] args) {

Scanner scanner = new Scanner(System.in);

int TestCases = scanner.nextInt();

for (int i = 0; i < TestCases; i++) {

int sizeOfArray = scanner.nextInt();

int Rotations = scanner.nextInt();

Rotations = Rotations % sizeOfArray;

scanner.nextLine();

String inputString = scanner.nextLine();

String[] inputStringArray = inputString.split(" ");

StringBuffer sb = new StringBuffer();

for (int j = 0; j < sizeOfArray; j++) {

sb.append(inputStringArray[(sizeOfArray + j - Rotations) % sizeOfArray] + " ");

}

System.out.print(sb);

System.out.println("");

}

}

}

**2.Monk and Inversions**

**URL**: [**https://www.hackerearth.com/problem/algorithm/monk-and-inversions-arrays-strings-e5aaa427/**](https://www.hackerearth.com/problem/algorithm/monk-and-inversions-arrays-strings-e5aaa427/)

**Solution**:

import java.util.\*;

class TestClass {

public static void main(String args[] ) throws Exception {

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

for(int a=0;a<T;a++){

int N = sc.nextInt();

int count = 0;

int[][] A= new int[N][N];

for(int i=0;i<N;i++){

for(int j=0;j<N;j++){

A[i][j] = sc.nextInt();

}

}

for(int i=0;i<N;i++){

for(int j=0;j<N;j++){

for(int k=i;k<N;k++){

for(int m=j;m<N;m++){

if(A[i][j]>A[k][m]){

count++;

}

}

}

}

}

System.out.print(count);

System.out.println();

}

}

}

**3.Cyclic shift**

**URL**: [**https://www.hackerearth.com/problem/algorithm/maximum-binary-number-cb9a58c1/**](https://www.hackerearth.com/problem/algorithm/maximum-binary-number-cb9a58c1/)

**Solution:**

import java.io.\*;

import java.util.\*;

class TestClass

{

static int compare(LinkedList<Character> A, LinkedList<Character> B){

Iterator<Character> i = A.iterator();

Iterator<Character> j = B.iterator();

if(A.size() == 0){ return -1;}

while (i.hasNext()) { // we know they have same length

char c = i.next();

char d = j.next();

if (c < d)

return -1;

else if (c > d)

return 1;

}

return 0;

}

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

while(T-- > 0){

int N; int K;

N = sc.nextInt();

K = sc.nextInt();

String input = sc.next();

LinkedList<Character> B = new LinkedList<>();

int d = 0;

int period = -1;

LinkedList<Character> inter = new LinkedList<>();

for(char c: input.toCharArray()){inter.add(c);}

for(int i = 0; i < N;i++){

if (compare(B, inter) < 0){

B = new LinkedList<>(inter);

d = i;

}else if (compare(B, inter) == 0){

period = i - d;

break;

}

inter.add(inter.removeFirst());

}

if(period == -1){

System.out.println(d + (K - 1L ) \* N);

}else{

System.out.println(d + ((K - 1L) \* period));

}

}

}

}

**4.Minimum AND xor OR**

**URL**: [**https://www.hackerearth.com/practice/data-structures/arrays/1-d/practice-problems/algorithm/minimum-and-xor-or-6a05bbd4/**](https://www.hackerearth.com/practice/data-structures/arrays/1-d/practice-problems/algorithm/minimum-and-xor-or-6a05bbd4/)

**Solution:**

import java.util.\*;

class TestClass {

public static void main(String args[] ) throws Exception {

Scanner sc = new Scanner(System.in);

int t = sc.nextInt();

while(t>0) {

t--;

int n = sc.nextInt();

int[] a = new int[n];

for(int i=0;i<n;i++) {

a[i] = sc.nextInt();

}

Arrays.sort(a);

int minium = a[n-1];

for(int i=n-1;i>0;i--) {

if(minium > (a[i]^a[i-1])) {

minium = a[i]^a[i-1];

}

}

System.out.println(minium);

}

sc.close();

}

}

**Sorting**

**1.Monk and Nice Strings**

**URL** : [**https://www.hackerearth.com/practice/algorithms/sorting/insertion-sort/practice-problems/algorithm/monk-and-nice-strings-3/**](https://www.hackerearth.com/practice/algorithms/sorting/insertion-sort/practice-problems/algorithm/monk-and-nice-strings-3/)

**Solution:**

import java.util.\*;

class TestClass{

public static void main(String args[]) throws Exception{

Scanner sc=new Scanner(System.in);

int n= sc.nextInt();

String A[]=new String[n];

int count;

for(int i=0;i<n;i++)

{

A[i]=sc.next();

}

System.out.println("0");

for(int i=1;i<n;i++)

{

count=0;

for(int j=0;j<i;j++){

if(A[i].compareTo(A[j])>0)

{

count++;

}

}

System.out.println(count);

}

}

}

**2.Monk and Suffix Sort**

**URL:** [**https://www.hackerearth.com/problem/algorithm/monk-and-suffix-sort-ebacdaf5/**](https://www.hackerearth.com/problem/algorithm/monk-and-suffix-sort-ebacdaf5/)

**Solution:**

import java.util.\*;

import java.lang.\*;

class TestClass{

public static void main(String args[]) throws Exception{

Scanner scanner=new Scanner(System.in);

String str=scanner.next();

int num =scanner.nextInt();

String strArray[] = new String[str.length()];

for(int i=0;i<str.length();i++)

{

strArray[i]=str.substring(i);

}

Arrays.sort(strArray);

System.out.println(strArray[num-1]);

}

}

**3.Monk being Monitor**

**URL**:

[**https://www.hackerearth.com/problem/algorithm/monk-being-monitor-709e0fd3/**](https://www.hackerearth.com/problem/algorithm/monk-being-monitor-709e0fd3/)

**Solution:**

import java.util.\*;

import java.lang.\*;

class TestClass {

public static void main(String args[] ) throws Exception

{

Scanner sc = new Scanner(System.in);

int t= sc.nextInt();

while(t --> 0){

int size= sc.nextInt();

int []A=new int[size];

for(int i=0;i<size;i++){

A[i]=sc.nextInt();

}

int current\_frequency = 0;

int min = 2147483647;

int result=0;

Arrays.sort(A);

for(int i=0;i<size;i++){

if(i!=size-1 && A[i]==A[i+1]){

current\_frequency++;

}

else{

current\_frequency++;

if(min>current\_frequency){

min=current\_frequency;

}

else{

result=Math.max(result,current\_frequency-min);

}

current\_frequency=0;

}

}

if(result>0){

System.out.println(result);

}

else{

System.out.println(-1);

}

}

}

}