***Hackerrank Algorithms***

**Time in Words:**

<https://www.hackerrank.com/challenges/the-time-in-words/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class Solution {

// Complete the timeInWords function below.

static String timeInWords(int h, int m)

{

String time="";

String a[]={"one","two","three","four","five","six","seven","eight","nine","ten","eleven","twelve"};

String b[]={"one","two","three","four","five","six","seven","eight","nine","ten","eleven","twelve","thirteen","fourteen","quarter","sixteen","seventeen","eighteen","nineteen","twenty","twenty one","twenty two","twenty three","twenty four","twenty five","twenty six","twenty seven","twenty eight","twenty nine",};

if(h<1 || h>12)

System.exit(0);

if(m<0 || m>59)

System.exit(0);

if(h==12 && m>0)

System.exit(0);

if(m==0)

{

time=a[h-1]+" o' clock";

}

else

{

if(m<30)

{

if(m==1)

{

time=b[m-1]+" minute past "+a[h-1];

}

else if(m==15)

{

time=b[m-1]+" past "+a[h-1];

}

else

{

time=b[m-1]+" minutes past "+a[h-1];

}

}

else if(m==30)

{

time="half past "+a[h-1];

}

else

{

if(m==59)

{

time=b[60-m-1]+" minute to "+a[h];

}

else if(m==45)

{

time=b[60-m-1]+" to "+a[h];

}

else

{

time=b[60-m-1]+" minutes to "+a[h];

}

}

}

return time;

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int h = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

int m = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

String result = timeInWords(h, m);

bufferedWriter.write(result);

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**3D Surface Area**

<https://www.hackerrank.com/challenges/3d-surface-area/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class Solution {

// Complete the surfaceArea function below.

static int surfaceArea(int[][] A,int H,int W)

{

int area=0;

if(H<1 || H>100 || W<1 || W>100)

System.exit(0);

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

{

for(int j=0;j<W;j++)

{

if(A[i][j]<1 || A[i][j]>100)

System.exit(0);

}

}

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

{

area+=A[0][i];

}

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

{

area+=A[i][0];

}

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

{

area+=A[H-1][i];

}

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

{

area+=A[i][W-1];

}

area+=2\*H\*W;

if(H>1 || W>1)

{

for(int i=W-1;i>=0;i--)

{

for(int j=H-1;j>=1;j--)

{

area+=Math.abs(A[j][i]-A[j-1][i]);

}

}

}

if(H>1 || W>1)

{

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

{

for(int j=0;j<W-1;j++)

{

area+=Math.abs(A[i][j]-A[i][j+1]);

}

}

}

return area;

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String[] HW = scanner.nextLine().split(" ");

int H = Integer.parseInt(HW[0]);

int W = Integer.parseInt(HW[1]);

int[][] A = new int[H][W];

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

String[] ARowItems = scanner.nextLine().split(" ");

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

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

int AItem = Integer.parseInt(ARowItems[j]);

A[i][j] = AItem;

}

}

int result = surfaceArea(A,H,W);

bufferedWriter.write(String.valueOf(result));

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Greedy Florist**

<https://www.hackerrank.com/challenges/greedy-florist/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class GreedyFlorist {

// Complete the getMinimumCost function below.

static int getMinimumCost(int k, int[] c)

{

int n=c.length;

if(n<1 || n>100 || k<1 ||k>100)

System.exit(0);

int cost=0;

int d=0,e=1;

GreedyFlorist gf=new GreedyFlorist();

gf.sort(c,0,n-1);

for(int i=n-1;i>n-k-1;i--)

{

cost+=c[i];

}

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

{

if(d==k)

{

d=0;

e++;

}

cost+=(e+1)\*c[i];

d++;

}

return cost;

}

void merge(int arr[], int l, int m, int r)

{

int n1 = m - l + 1;

int n2 = r - m;

int L[] = new int [n1];

int R[] = new int [n2];

for (int i=0; i<n1; ++i)

L[i] = arr[l + i];

for (int j=0; j<n2; ++j)

R[j] = arr[m + 1+ j];

int i = 0, j = 0;

int k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

/\* Copy remaining elements of L[] if any \*/

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

/\* Copy remaining elements of R[] if any \*/

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void sort(int arr[], int l, int r)

{

if (l < r)

{

// Find the middle point

int m = (l+r)/2;

// Sort first and second halves

sort(arr, l, m);

sort(arr , m+1, r);

// Merge the sorted halves

merge(arr, l, m, r);

}

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String[] nk = scanner.nextLine().split(" ");

int n = Integer.parseInt(nk[0]);

int k = Integer.parseInt(nk[1]);

int[] c = new int[n];

String[] cItems = scanner.nextLine().split(" ");

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

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

int cItem = Integer.parseInt(cItems[i]);

c[i] = cItem;

}

int minimumCost = getMinimumCost(k, c);

bufferedWriter.write(String.valueOf(minimumCost));

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Max Min**

<https://www.hackerrank.com/challenges/angry-children/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class MinMax

{

// Complete the maxMin function below.

static int maxMin(int k, int[] arr)

{

int n=arr.length;

int a[]=new int[n+1];

MinMax mm=new MinMax();

mm.sort(arr,0,arr.length-1);

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

{

a[i]=arr[i];

}

a[n]=arr[0];

int min=Integer.MAX\_VALUE;

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

{

int t[]=new int[k];

int d=0;

for(int j=i;j<i+k;j++)

{

t[d]=a[j];

d++;

}

int x=maxx(t)-minn(t);

if(x<=min)

min=x;

}

return min;

}

static int maxx(int t[])

{

int max=t[0];

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

{

if(max<t[i])

max=t[i];

}

return max;

}

static int minn(int t[])

{

int min=t[0];

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

{

if(min>t[i])

min=t[i];

}

return min;

}

void merge(int arr[], int l, int m, int r)

{

int n1 = m - l + 1;

int n2 = r - m;

int L[] = new int [n1];

int R[] = new int [n2];

for (int i=0; i<n1; ++i)

L[i] = arr[l + i];

for (int j=0; j<n2; ++j)

R[j] = arr[m + 1+ j];

int i = 0, j = 0;

int k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

/\* Copy remaining elements of L[] if any \*/

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

/\* Copy remaining elements of R[] if any \*/

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void sort(int arr[], int l, int r)

{

if (l < r)

{

// Find the middle point

int m = (l+r)/2;

// Sort first and second halves

sort(arr, l, m);

sort(arr , m+1, r);

// Merge the sorted halves

merge(arr, l, m, r);

}

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int n = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

int k = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

int[] arr = new int[n];

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

int arrItem = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

arr[i] = arrItem;

}

int result = maxMin(k, arr);

bufferedWriter.write(String.valueOf(result));

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Fibonacci Modified**

<https://www.hackerrank.com/challenges/fibonacci-modified/problem>

import java.io.\*;

import java.math.\*;

import java.math.BigInteger;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class ModifiedFibo

{

// Complete the fibonacciModified function below.

static BigInteger fibonacciModified(int t1, int t2, int n)

{

BigInteger a=BigInteger.valueOf(t1);

BigInteger b=BigInteger.valueOf(t2);

int dd=0;

BigInteger c=BigInteger.valueOf(dd);

BigInteger d=BigInteger.valueOf(dd);

for(int i=3;i<=n;i++)

{

d=b.multiply(b);

c=a.add(d);

a=b;

b=c;

}

return c;

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String[] t1T2n = scanner.nextLine().split(" ");

int t1 = Integer.parseInt(t1T2n[0]);

int t2 = Integer.parseInt(t1T2n[1]);

int n = Integer.parseInt(t1T2n[2]);

//int result = fibonacciModified(t1, t2, n);

bufferedWriter.write(String.valueOf(fibonacciModified(t1, t2, n)));

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Encryption**

<https://www.hackerrank.com/challenges/encryption/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class Encrypt{

static String remove(String s)

{

int n=s.length();

String temp="";

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

{

if(s.charAt(i)!=' ')

temp+=s.charAt(i);

}

return temp;

}

// Complete the encryption function below.

static String encryption(String s)

{

String k=remove(s);

int n=k.length();

int a=(int)(Math.floor(Math.sqrt(n)));

int b=(int)(Math.ceil(Math.sqrt(n)));

if((a\*b)<n)

{

a=a+1;

b=b+1;

int c;

c=a;

a=b;

b=c;

char grid[][]=new char[a][b];

int x=0;

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

{

for(int j=0;j<b;j++)

{

if(x!=n)

{

grid[i][j]=k.charAt(x);

x++;

}

else

break;

}

}

String temp="";

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

{

for(int j=0;j<a;j++)

{

if(grid[j][i]!='\u0000')

temp+=grid[j][i];

}

temp+=" ";

}

return temp;

}

else

{

char grid[][]=new char[a][b];

int x=0;

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

{

for(int j=0;j<b;j++)

{

if(x!=n)

{

grid[i][j]=k.charAt(x);

x++;

}

else

break;

}

}

String temp="";

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

{

for(int j=0;j<a;j++)

{

if(grid[j][i]!='\u0000')

temp+=grid[j][i];

}

temp+=" ";

}

return temp;

}

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String s = scanner.nextLine();

String result = encryption(s);

bufferedWriter.write(result);

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Bigger is Greater**

<https://www.hackerrank.com/challenges/bigger-is-greater/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class Solution {

// Complete the biggerIsGreater function below.

static String biggerIsGreater(String w)

{

int i;

char ch[]=new char[w.length()];

for(int k=0;k<w.length();k++)

{

ch[k]=w.charAt(k);

}

int n=ch.length;

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

{

if (ch[i]>ch[i-1])

{

break;

}

}

String temp="";

if(i==0)

{

temp="no answer";

}

else

{

int x=ch[i-1],min=i;

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

{

if (ch[j]>x && ch[j]<ch[min])

{

min=j;

}

}

char tmp = ch[i-1];

ch[i-1] = ch[min];

ch[min] = tmp;

Arrays.sort(ch,i,n);

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

temp+=ch[i];

}

return temp;

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int T = scanner.nextInt();

scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

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

String w = scanner.nextLine();

String result = biggerIsGreater(w);

bufferedWriter.write(result);

bufferedWriter.newLine();

}

bufferedWriter.close();

scanner.close();

}

}

**Sherlock and the Valid String**

<https://www.hackerrank.com/challenges/sherlock-and-valid-string/problem>

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.regex.\*;

public class SherlockString {

// Complete the isValid function below.

static String isValid(String s)

{

int n=s.length();

String k=s.toLowerCase();

String temp="";

int a[]=new int[26];

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

{

int x=(int)k.charAt(i);

a[x-97]++;

}

int j=a[0];

int o=j+1;

int l=0;

int m=0;

for(int i=0;i<26;i++)

{

if(a[i]!=j && a[i]!=0)

{

l=l+1;

}

else if(a[i]!=o)

{

m=m+1;

}

if(l==0 || m==25)

temp="YES";

else

temp="NO";

}

System.out.println(m+"\n");

for(int i=0;i<26;i++)

System.out.println(a[i]);

return temp;

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String s = scanner.nextLine();

String result = isValid(s);

bufferedWriter.write(result);

bufferedWriter.newLine();

bufferedWriter.close();

scanner.close();

}

}

**Powers Game**

<https://www.hackerrank.com/challenges/powers-game-1/problem>

import java.io.\*;

import java.math.\*;

import java.text.\*;

import java.util.\*;

import java.util.regex.\*;

public class Solution {

/\*

\* Complete the powersGame function below.

\*/

static String powersGame(int n)

{

if(n%8==0)

return "Second";

else

return "First";

}

private static final Scanner scanner = new Scanner(System.in);

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

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

int t = Integer.parseInt(scanner.nextLine().trim());

for (int tItr = 0; tItr < t; tItr++) {

int n = Integer.parseInt(scanner.nextLine().trim());

String result = powersGame(n);

bufferedWriter.write(result);

bufferedWriter.newLine();

}

bufferedWriter.close();

}

}

**Zig Zag Sequence**

<https://www.hackerrank.com/challenges/zig-zag-sequence/problem>

import java.util.\*;

import java.lang.\*;

import java.io.\*;

import java.math.\*;

public class Main {

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

Scanner kb = new Scanner(System.in);

int test\_cases = kb.nextInt();

for(int cs = 1; cs <= test\_cases; cs++){

int n = kb.nextInt();

int a[] = new int[n];

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

a[i] = kb.nextInt();

}

findZigZagSequence(a, n);

}

}

public static void findZigZagSequence(int [] a, int n){

Arrays.sort(a);

int mid = (n + 1)/2-1;

int temp = a[mid];

a[mid] = a[n - 1];

a[n - 1] = temp;

int st = mid + 1;

int ed = n - 2;

while(st <= ed){

temp = a[st];

a[st] = a[ed];

a[ed] = temp;

st = st + 1;

ed = ed - 1;

}

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

if(i > 0) System.out.print(" ");

System.out.print(a[i]);

}

System.out.println();

}

}

Prime Dates

<https://www.hackerrank.com/challenges/prime-date/problem>

import java.util.\*;

import java.lang.\*;

import java.io.\*;

import java.math.\*;

public class Main {

public static int month[];

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

Scanner in = new Scanner(System.in);

month = new int[15];

String s = in.nextLine();

StringTokenizer str = new StringTokenizer(s, "- ");

int d1 = Integer.parseInt(str.nextToken());

int m1 = Integer.parseInt(str.nextToken());

int y1 = Integer.parseInt(str.nextToken());

int d2 = Integer.parseInt(str.nextToken());

int m2 = Integer.parseInt(str.nextToken());

int y2 = Integer.parseInt(str.nextToken());

int result = findPrimeDates(d1, m1, y1, d2, m2, y2);

System.out.println(result);

}

public static void updateLeapYear(int year) {

if(year % 400 == 0) {

month[2] = 29;

} else if(year % 100 == 0) {

month[2] = 28;

} else if(year % 4 == 0) {

month[2] = 29;

} else {

month[2] = 28;

}

}

public static void storeMonth() {

month[1] = 31;

month[2] = 28;

month[3] = 31;

month[4] = 30;

month[5] = 31;

month[6] = 30;

month[7] = 31;

month[8] = 31;

month[9] = 30;

month[10] = 31;

month[11] = 30;

month[12] = 31;

}

public static int findPrimeDates(int d1, int m1, int y1, int d2, int m2, int y2) {

storeMonth();

int result = 0;

while(true) {

int x = d1;

x = x \* 100 + m1;

x = x \* 10000 + y1;

if(x % 4 == 0 || x % 7 == 0) {

result = result + 1;

}

if(d1 == d2 && m1 == m2 && y1 == y2) {

break;

}

updateLeapYear(y1);

d1 = d1 + 1;

if(d1 > month[m1]) {

m1 = m1 + 1;

d1 = 1;

if(m1 > 12) {

y1 = y1 + 1;

m1 = 1;

}

}

}

return result;

}

}