**MIU Preliminary Open contest #2**

## **A - Utopian Tree**

**Solution in c:**

#include <math.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <assert.h>

#include <limits.h>

#include <stdbool.h>

int main(){

int t,x=1,a[10];

scanf("%d",&t);

for(int a0 = 0; a0 < t; a0++){

int n;

scanf("%d",&n);

if(n==0)

a[a0]=x;

else if(n%2==0)

{

while(n!=0){

x=2\*x+1;

n-=2;

}

}

else{

while(n!=1){

x=2\*x+1;

n-=2;

}

if(n==1){

x=2\*x;

}

}

a[a0]=x;

x=1;

}

for(int a0 = 0; a0 < t; a0++){

printf("%d\n",a[a0]);

}

return 0;

}

C - Mother bear

**Solution in java:**

import java.io.\*;

class Main {

public static void main(String... args) throws IOException {

InputStreamReader isr = new InputStreamReader(System.in);

BufferedReader in = new BufferedReader(isr);

String line;

while (!(line = in.readLine()).equals("DONE")) {

line = line.replaceAll("[\\p{Punct} ]", "");

line = line.toLowerCase();

boolean valid = true;

for (int i = 0; i < line.length() / 2 && valid; i++) {

valid = line.charAt(i) == line.charAt(line.length() - i - 1);

}

System.out.println(valid ? "You won't be eaten!" : "Uh oh..");

}

}

}

## **D - Ordering Tasks**

**Solution in java:**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.StringTokenizer;

class Main {

public static int [][] edges;

public static int [] edgesCount;

public static boolean [] visited;

public static int [] ints;

public static int intsCount;

public static void topologicalSort (int id) {

if (!visited[id]) {

visited[id]=true;

for (int i=0;i<edgesCount[id];i++) {

topologicalSort(edges[id][i]);

}

ints[intsCount++]=id;

}

}

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

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

while (true) {

StringTokenizer st=new StringTokenizer(br.readLine());

int n=Integer.parseInt(st.nextToken());

int m=Integer.parseInt(st.nextToken());

if (n==0 && m==0) {

break;

}

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

edgesCount=new int [n+1];

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

st=new StringTokenizer(br.readLine());

int src=Integer.parseInt(st.nextToken());

int dest=Integer.parseInt(st.nextToken());

edges[src][edgesCount[src]++]=dest;

}

visited=new boolean [n+1];

ints=new int [n];

intsCount=0;

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

topologicalSort(i);

}

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

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

if (i==0) {

System.out.println();

} else {

System.out.print(" ");

}

}

}

}

}

E - Angry Professor

**Solution in c:**

#include <math.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <assert.h>

#include <limits.h>

#include <stdbool.h>

int main(){

int t;

scanf("%d",&t);

for(int a0 = 0; a0 < t; a0++){

int n;

int k;

int count=0;

scanf("%d %d",&n,&k);

int a[n];

for(int a\_i = 0; a\_i < n; a\_i++){

scanf("%d",&a[a\_i]);

}

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

{

if(a[i]<=0) count++;

}

if(count>=k)

printf("NO \n");

else

printf("YES \n");

}

return 0;

}

G - Chocolate Feast

**Solution in java:**

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

/\* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. \*/

Scanner input = new Scanner(System.in);

int t = input.nextInt();

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

{

int n = input.nextInt();

int c = input.nextInt();

int m = input.nextInt();

int ate = 0;

int chocolates = n / c;

ate += chocolates;

while(chocolates >= m)

{

ate += chocolates / m;

chocolates = (chocolates / m) + (chocolates % m);

}

System.out.println(ate);

}

}

}

H - Drawing Book

Solution in java:

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

public class Solution {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int n = in.nextInt();

int p = in.nextInt();

if(p==n||p==1)

{

System.out.println('0');

}// your code goes here

else if(p<=(n/2))

{

System.out.println(p/2);

}

else if(p>(n/2))

{

if(n%2==0)

{

System.out.println(((n-p)+1)/2);

}

else

{

System.out.println(((n-p))/2);

}

}

}

}

I - Coin Collector

**Solution in java:**

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.LineNumberReader;

import java.math.BigInteger;

import java.util.\*;

public class Main {

public static int TestCases;

public static long All;

public static int Index, Answer;

public static int NumberOfCoins;

public static int[] List;

public static int[] Values;

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

Scanner Input = new Scanner(System.in);

TestCases = Input.nextInt();

int Cases = 1;

while(TestCases-- > 0) {

All = Index = Answer = 0;

NumberOfCoins = Input.nextInt();

List = new int[NumberOfCoins + 1];

Values = new int[NumberOfCoins + 1];

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

List[i] = Input.nextInt();

}

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

while(All >= List[i]) All -= Values[--Index];

All += List[i];

Values[Index++] = List[i];

if(Index > Answer) Answer = Index;

}

System.out.println(Answer);

}

}

}

J - The Bus Driver Problem

Solution in java:

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.\*;

public class Main {

public static int NumberOfBusDrivers, Distance, Over;

public static int[] Morning;

public static int[] Evening;

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

Scanner Input = new Scanner(System.in);

while(true){

NumberOfBusDrivers = Input.nextInt();

Distance = Input.nextInt();

Over = Input.nextInt();

if(NumberOfBusDrivers == 0 && Distance == 0 && Over == 0) break;

Morning = new int[NumberOfBusDrivers];

Evening = new int[NumberOfBusDrivers];

for(int i = 0; i < NumberOfBusDrivers; i++) Morning[i] = Input.nextInt();

for(int i = 0; i < NumberOfBusDrivers; i++) Evening[i] = -Input.nextInt();

Arrays.sort(Morning); Arrays.sort(Evening);

int Value = 0;

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

int Result = Morning[i] + (-Evening[i]);

if(Result > Distance)

Value += (Result - Distance) \* Over;

}

System.out.println(Value);

}

}

}