**Algorithmic Problem Solving [17ECSE309]**

**Q-Box Assignment Set**

Student Name: Shreyas Joshi

SRN: 01fe20bcs145

Branch: CS

**Question 02**

Title:

Level: Medium

Concepts Tested:

**Problem Statement:**

ShreeHari is running a lottery business where he sells lottery tickets every day for a certain number of days. Each ticket has a number, and there is a group of people who buy these tickets every day. The number of winners in the lottery is determined by a fixed key value (k) that is the same for all days. If the binary representation of a ticket number has exactly k digits, then the person holding that ticket is a winner.

The more winners there are on a particular day, the more money ShreeHari loses. Your task is to help ShreeHari find the day on which he loses the most money. To do this, you need to know the number of days (m), the number of people buying tickets each day (n), the fixed key value (k), and the ticket numbers for each day(t).

**Input Format:**

The first line contains a single integer (m), which represents the number of days for which lottery tickets are sold.

The second line contains a single integer (n), which represents the number of people buying lottery tickets each day.

The third line contains a single integer (k), which is the fixed key value that determines the number of digits in the binary representation of the winning ticket number.

The fourth line contains m \* n space-separated integers that represent the ticket numbers sold on each day.

**Constraints:**

1<=m<=4294967295

1<=n<=4294967295

0<=k<=64

0<=t<=9223372036854775807

**Output Format:**

A single integer denoting the day on which ShreeHari loses the most money. If there is no day on which ShreeHari loses money, then print 0. If multiple days have the same number of winners, leading to the same loss, then you should print the first day among those days.

**Solution:**

*#include*<bits/stdc++.h>

using namespace std;

int main()

{

*//m indicates days*

*//n indicates number of people who bought ticket on each day*

*//vec[i][j] indicates ticket ID*

*//k indicates key*

    int m,n;

    cin>>m;

    cin>>n;

    int k;

    cin>>k;

    vector<vector<long long int>> vec( m , vector<long long int> (n, 0));

*//Take input*

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

    {

*for*(int j=0;j<n;j++)

        {

            cin>>vec[i][j];

        }

    }

    int newvalue;

    vector<long long int> answer(m,0);

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

    {

*for*(int j=0;j<n;j++)

        {

            newvalue=log2(vec[i][j])+1; *//This is logic to get nnumber of digits in binary representation in constant time.*

*if*(newvalue == k)

            {

                answer[i]++; *//Increment the loss count*

            }

        }

    }

*//To find miximum loss*

    int maxi=answer[0];

    int maxind=0;

*for*(int i=1;i<m;i++)

    {

*if*(answer[i]>maxi)

        {

            maxi=answer[i];

            maxind=i;

        }

    }

*if*(maxi==0)

    {

        cout<<"No loss"<<endl;

    }

*else*

        cout <<maxind+1;

*return* 0;

}

**Sample Test Cases:**

**Sample Input 0**

2

2

4

21 8 9 10

**Sample Output 0**

**2**

**Explanation 0**

**Two** individuals purchased tickets for a **two**-day event. On the **first** day, the ticket numbers were **21** and **8**. However, the number **8** had a binary representation consisting of four digits, resulting in one loss incurred. On the second day, the ticket numbers were **9** and **10**, and both of them had binary representations consisting of **four** digits, resulting in two losses incurred. The highest number of losses were incurred on day **2**. Therefore, the final output is 2.

**Sample Input 1**

2

3

9

34 98 3411 89 11 0

**Sample Output 1**

No loss

**Explanation 1**

There are **3** people and tickets are sold **for 2** days.

On day one there are no ticket numbers, such that, the number of digits in its binary representation is equal to **9**. Therefore he incurred zero loss on day **1**.

Similarly on day two, he incurred zero loss. Therefore the final output is **No loss**.

Other test cases are in text files

**Question 01**

Title:

Level: Easy

Concepts Tested:

**Problem Statement:**

Ram enjoys walking and wants to cover the maximum distance possible by walking in the parks. He has a list of n parks, where each park has a race track of a certain length given in kilometers. However, he is only allowed to walk in k parks per day. We need to find the maximum distance he can cover by walking in the parks, given this restriction.

**InputFormat:**

First line contains n.

Next n line contains distance of race track of n parks.

Last line contains k

**Constraints:**

1<=n<=2^32

1<=arr[i]<=2^32

1<=k<=n

total

**Output Format:**

A single integer value denoting the maximum distance covered by Ram.

**Solution:**

*#include*<bits/stdc++.h>

using namespace std;

int main()

{

    int n;

    cin>>n;

    vector<int>v(n);

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

    {

        cin>>v[i];

    }

    int k;

    cin>>k;

    sort(v.begin(),v.end(),greater<int>());

    int answer=0;

    int i=0;

*while*(k--)

    {

        answer+=v[i];

        i++;

    }

    cout<<answer<<endl;

}

**Sample Test Cases:**

**Sample Test Cases 0**

**Input 0**

5

1

3

11

221

77

2

**Output 0**

298

**Explanation 0**

There are 5 parks and Ram is restricted to walk in 2 parks. He choose 2 parks with distance 221 and 77 therefore maximum distance covered is 298.

**Sample Test Cases 1**

**Input 1**

11

111

31

91

228

775

29

345

6456

34

556

656

5

**Output 1**

8788

**Explanation 1**

There are 11 parks and Ram is restricted to walk in 5 parks. He choose 5 parks with distance 6456, 775, 656, 556 and345 therefore maximum distance covered is 8788.

Other test cases are txt files.