Judging Director: Abu Saleh
Associate Judging Director: Masum Bin Hossain
Slot D Judging Director: Md. Galib Hossain
Slot D Associate Judging Director: Mohammad Dipo Sultan

A. Flying Horse

Category: Give away

Problem Setter: Anamika Hossain Lily
Reviewer: Syed Ahsanul Huque Nahid

Analysis:

Just have to print a single line **Ada Lovelace,The Enchantress of Numbers!**

Code:

```
#include<stdio.h>

int main()

{

    printf("Ada Lovelace, The Enchantress of Numbers!\n");

    return 0;

}
```

B. Hatred

Category: Simple If-Else

Problem Setter: Ariful Hoque
Reviewer: Anamika Hossain Lily

Analysis:

All You’ve to do is check whether the input value is equal to 666666 or not. If yes, print “Agh! It opened” else print “Aww! You failed to open”

Code:

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

using namespace std;

int main()

{

   string s;

   cin>>s;

   if (s=="666666")

   {

      printf("Agh! It opened");

   }

   else

   {

      printf("Aww! You failed to open");

   }

   return 0;

}
```

C. Clumsy Scoreboard

Category: Basic math

Problem Setter: Imrul Hasan Sifat
Reviewer: Ariful Hoque
Analysis:

Basically, the problem asks to find the **minimum number of runs** required by the Bangladesh team to win the match.
Two pieces of information are given, one is the score India scored in their first innings (X) and the current score of the Bangladesh team(Y). So, simply the Bangladesh team just needed to score one more run than the India team to win the match.

Solution:
X = Runs scored by India
Y = Current runs scored by Bangladesh
Minimum runs needed to score for Bangladesh to win the match:

$$(X - Y) + 1$$

Code:

```
#include<stdio.h>

int main()

{

    int x, y;

    scanf("%d %d", &x, &y);

    printf("%d", (x-y)+1);


    return 0;


}
```

D. G.O.A.T

Category: Nested if-else/Easy Implementation

Problem Setter: Mohammad Dipo Sultan
Reviewer: Imrul Hasan Sifat

Analysis:

This problem can be solved using nested if else . We need to check the given condition. Here our first priority is goal, between two of them who score more goals he will be the winner. If goals are the same then we check for assists, if any of them score more assists will be the winner. If the goal and assist count is also same then CR10 will be the winner.

Solution: These conditions can be handled by using if-else.

Code:

```
#include<stdio.h>
int main()
{
   int A,B,C,D;
   scanf("%d%d%d%d", &A,&B, &C,&D);
   if(A>C)
   {
      printf("GOAT CR10");
   }
   else if(C>A)
   {

      printf("GOAT LM7");
   }

   else if(A==C)
   {
      if(B>D)
      {
         printf("GOAT CR10");
      }
```

```
        else if (D>B)
        {
            printf("GOAT LM7");
        }
        else
        {
            printf("GOAT CR10");
        }
    }
    return 0;
}
```

E. Chocolate

Category: Loop

Problem Setter: Syed Ahsanul Huque Nahid
Reviewer: Mohammad Dipo Sultan
Special Thanks: Mohammad Dipo Sultan
Analysis:

In this problem, there is one observation. That we can’t take more than X chocolate. So we can say it is a simple summation problem with a loop except for the fact that if the value we are adding is greater than a certain value we can’t take more than that.
So, we can add all the values but if any value is greater than X then we should add X instead of that value.

Code:

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

using namespace std;
```

```
int main(){
        int n,k,sum=0;
        cin>>n>>k;
        while(n--){
                int x;
                cin>>x;
                sum+=min(x,k);
        }
        cout<<sum<<endl;
        return 0;
}
```

F. Meet in the Middle

Category: Loop

Problem Setter: Md. Galib Hossain
Reviewer: Syed Ahsanul Huque Nahid
Special Thanks: MD Albin Hossain, MD Nabid Anzum Akash, Sabbir Ahamed Mridha

Analysis:

Observation: We need to find the euclidean distance between any two points of the rectangle.

Solution Idea: If two points are adjacent then we need to find the distance between the two points. If they are not adjacent then the input can be either A->C or B->D then we have to find A->B then B->C or B->A then A->D.

Code:

```
#include <bits/stdc++.h>
using namespace std;
```

```
const double INF = 1e9;
const double EPS = 1e-9;

struct point {  double x, y;};

double dist(point p1, point p2) {  return hypot(p1.x-p2.x, p1.y-p2.y);}

int main() {
   point a[10];
   for (int i = 0; i < 4; i++) cin >> a[i].x >> a[i].y;

   vector<char> v(2);
   cin >> v[0] >> v[1];
   sort(v.begin(), v.end());
   double ans;
   if ((abs(v[1] - v[0])%3 )> 1) { ans = dist(a[v[0] - 'A'], a[v[0] - 'A' + 1]) + dist(a[v[1] - 'A'],
a[v[0] - 'A' + 1]);    }
   else ans = dist(a[v[0] - 'A'], a[v[1] - 'A']);

   printf("%.15lf\n", ans);

   return 0;
}
```

G. Doppelgänger

Category: String

Problem Setter: Mohammad Dipo Sultan
Reviewer: Syed Ahsanul Huque Nahid

Analysis:

The string which will contain Latin letters upper, lowercase and number 1 to 9. If we just split numbers and store those numbers in int, and with remaining characters form a new string. Then we will check from our new number which was concreted from the string, is this divisible by k which is given input in the problem.

Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    string s;
    cin>>n>>n>>s;
    string st="";
    int out=0;
    for (auto i:s)
    {
        if (i>='0'&&i<='9')
        {
            out=out*10+i-'0';
            st+=i;
            out%=n;
        }
    }
    bool f=1;
    while(st.size()!=1&&st[0]=='0')
    {
        st.erase(st.begin());
    }
    if (out)
    {
        for(auto i:s)
        {
            if (!(i>='0'&&i<='9'))cout<<i;
        }
        cout<<endl;
    }
    else cout<<st<<endl;

    return 0;
}
```

H. Take the Challenge!!

Category: Number Theory

Problem Setter: Syed Ahsanul Huque Nahid
Reviewer: Mohammad Dipo Sultan
Analysis:

In this problem first, we have to find the number of divisors over the array. If we do this in sqrt then the problem will lead to tle. So first we should learn to find the number of divisors in cube root. We can do that by prime factorization with sieve.

Then in the second part there is a simple observation. That is, if we take the maximum attack power from the new array only one time then our average will be maximized. if we take any other number with it then the average will decrease or stay equal. so for this part we need to find out the maximum number in the array. and finally print the maximum number.

Code:

solution1-

```
#include <bits/stdc++.h>
using namespace std;
int nod(int n)
{
    int x=1;
    for(int i=2; i<=sqrt(n); i++)
    {
        int cnt{};
        if (n%i==0)
        {
            while(n%i==0)n/=i,cnt++;
        }
        x*=(cnt+1);
    }
    if (n>1)x*=2;
    return x;
}
int main()
{
    int t,cs{};
    cin>>t;
    while(t--)
    {
        int n;
        cin>>n;
        int ar[n];
        for(int&i:ar)cin>>i;
        for(int&i:ar)i=nod(i);
        int mx=-1;
        for(auto i:ar)mx=max(mx,i);
        printf("Case %d: %0.4lf\n",++cs,(double)mx);
    }
    return 0;
}
```

Solution2-

```
#include <bits/stdc++.h>
using namespace std;
vector<int>v;
void sieve(int n)
{
    bool prime[n + 1];
    memset(prime, true, sizeof(prime));
    for (int p = 2; p * p <= n; p++)
    {
        if (prime[p] == true)
        {
            for (int i = p * p; i <= n; i += p)
                prime[i] = false;
        }
    }
    for (int p = 2; p <= n; p++)
        if (prime[p])
            v.push_back(p);
}
int nod(int n)
{
    int x=1;
    for(int i=0; i<v.size()&&v[i]<=sqrt(n); i++)
    {
        int cnt{};
        if (n%v[i]==0)
        {
            while(n%v[i]==0)n/=v[i],cnt++;
        }
        x*=(cnt+1);
    }
    if (n>1)x*=2;
    return x;
}

int main()
{
    sieve(sqrt(1000000000));
    int t,cs{};
    cin>>t;
    while(t--)
    {
        int n;
        cin>>n;
        int ar[n];
        for(int&i:ar)cin>>i;
        for(int&i:ar)i=nod(i);
        int mx=-1;
        for(auto i:ar)mx=max(mx,i);
        printf("Case %d: %0.4lf\n",++cs,(double)mx);
    }
    return 0;
}
```