سوال A: 800

https://codeforces.com/problemset/problem/71/A

طبق خواسته سوال، اگر کلمه طول کمتر از 10 دارد، باید به همان صورت چاپ شود، در غیر این صورت ابتدا اولین کاراکتر رشته، سپس سایز رشته منهای 2 و سپس آخرین کاراکتر رشته را چاپ میکنیم .

C++:

```cpp
#include <iostream>
using namespace std;
int main(){
    string s;
    cin>>s;
    while(cin>>s){
        if(s.size()<11) cout<<s<<endl;
        else cout<<s[0]<<s.size()-2<<s[s.size()-1]<<endl;
    }
    return 0;
}
```

Python:

```python
for i in [0]*int(input()):
   w=input()
   l=len(w)-2
   print([w,w[0]+str(l)+w[-1]][l>8])
```

سوال B: 800

https://codeforces.com/problemset/problem/1557/A

از آنجایی که میانگین یک گروه از اعداد همیشه مقداری بین حداقل و حداکثر اعداد در آن گروه دارد، می توان ثابت کرد که بهترین رویکرد برای به دست آوردن حداکثر مجموع میانگین های دو دنباله فرعی، قرار دادن حداکثر عدد به تنهایی در یک زیر دنباله و بقیه اعداد را به دنباله دیگر است.

C++:

```cpp
#include<bits/stdc++.h>
using namespace std;
typedef long double ll;
int main(){
    ll t;
    cin>>t;
    while(t--)
    {
        int n;
        cin>>n;
        int ar[n];
        ll sum=0;
        for(int i=0;i<n;i++){
            cin>>ar[i];
            sum+=ar[i];
        }
        sort(ar,ar+n);
        cout<<fixed<<setprecision(10)<<(sum-ar[n-1])/(n-1)+ar[n-1]<<"\n";
    }
}
```

Python:

```python
t=int(input())
while(t>0):
   n=int(input())
   a=list(map(int,input().split()))
   print(max(a)+(sum(a)-max(a))/(n-1))
   t-=1
```

سوال C: 900

https://codeforces.com/problemset/problem/1272/A

این مشکل با شبیه سازی ساده قابل حل است. اجازه دهید $na \in \{a-1,a,a+1\}$ موقعیت جدید اولین دوست باشد، $nb \in \{b-1,b,b+1\}$ و $nc \in \{c-1,c,c+1\}$ به ترتیب موقعیت های جدید دوستان دوم و سوم هستند. برای موقعیت های ثابت می توانید پاسخ را با مقدار $|na-nb|+|na-nc|+|nb-nc|$ به دست آورید. و تکرار بیش از سه موقعیت را می توان با حلقه های تو در تو پیاده سازی کرد.

C++:

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

int calc(int a, int b, int c) {
    return abs(a - b) + abs(a - c) + abs(b - c);
}

int main() {
    int q;
    cin >> q;
    for (int i = 0; i < q; ++i) {
        int a, b, c;
        cin >> a >> b >> c;
        int ans = calc(a, b, c);
        for (int da = -1; da <= 1; ++da) {
            for (int db = -1; db <= 1; ++db) {
                for (int dc = -1; dc <= 1; ++dc) {
                    int na = a + da;
                    int nb = b + db;
                    int nc = c + dc;
                    ans = min(ans, calc(na, nb, nc));
                }
            }
        }
        cout << ans << endl;
    }
}
```

Python:

```python
q = int(input())
for i in range(q):
        a, b, c = map(int, input().split())
        ans = abs(a - b) + abs(a - c) + abs(b - c)
        for x in range(-1, 2):
            for y in range(-1, 2):
                for z in range(-1, 2):
                    na = a+x
                    nb = b+y
                    nc = c+z
                    ans = min(ans, abs(na - nb) + abs(na - nc) + abs(nb - nc))
        print( ans )
```

سوال D: 1000

https://codeforces.com/problemset/problem/1515/B

اگر n را بتوان x2 یا x4 نوشت که x یک عدد مربع کامل است، پاسخ برابر با YES خواهد بود و در غیر این صورت پاسخ برابر با NO خواهد بود.

برای تجسم این ساختار، ابتدا با ساختن یک مربع کوچکتر با استفاده از 2 یا 4 قطعه شروع می کنیم (روش ساخت مربع با 2 و یا 4 قطعه در سوال ترسیم شده اند). برای ساخت یک مربع واحد از قطعه های تشکیل شده لازم است که x قطعه از مربع ها را در کنار یکدیگر بگذاریم(به همین دلیل x باید مربع کامل باشد).

C++:

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

bool isSquare(int x){
    int y=sqrt(x);
    return y*y==x;
}

void solve(){
    int n;
    cin>>n;
    if (n%2==0 && isSquare(n/2))
        cout<<"YES"<<endl;
    else if (n%4==0 && isSquare(n/4))
        cout<<"YES"<<endl;
    else
        cout<<"NO"<<endl;
}

int main(){
    int t; cin>>t;
    while (t--)
        solve();
}
```

Python:

```python
import math
for t in range(int(input())):
    n = int(input())
    a = math.sqrt(n/2)
    b = math.sqrt(n/4)
    if round(a) == a or round(b) == b:
        print("YES")
    else:
        print("NO")
```

سوال E: 1500

https://codeforces.com/problemset/problem/1474/B

از آنجایی که آرایه a دارای اعداد فرد و آرایه b دارای اعداد زوج است، مهم نیست که چگونه عملیات را انجام دهیم، همواره دو خانه اول دو متفاوت خواهند بود. یعنی همه چیز به دو خانه اول بستگی دارد. نتیجه این است که برای اینکه آرایه اول از نظر lexicographically کوچکتر از آرایه دوم باشد، باید اولین خانه a را از خانه اول b کوچکتر کنیم. برای انتقال خانه i ام یک آرایه به موقعیت اول، می توانیم با انجام عملیات بر روی خانه i-1 ام و سپس خانه i-2 ام و به همین ترتیب تا خانه اول، عنصر موجود در خانه i ام را به موقعیت اول برسانیم.

ابتدا باید بدانیم که لازم است برای هر bi که بخواهیم آن را به خانه اول برسانیم (مشخص است که هزینه این کار برابر با i است) کمترین هزینه ای که باید مصرف کنیم تا aj ای را که کوچکتر از bi باشد را به خانه اول برسانیم چقدر است. در صورت انجام این کار به صورت عادی با مشکل TLE مواجه خواهیم شد. برای حل این مشکل باید میتوان از روش زیر استفاده کرد.

فرض کنیم که مقدار x را قرار است به خانه اول b برسانیم، اگر بتوانیم در زمان مناسب مشخص کنیم که برای هر مقدار x که در خانه اول b قرار میگیرد، کمترین هزینه ای که می توانیم پرداخت کنیم تا خانه اول a کوچکتر از خانه اول b شود چقدر است، مسئله به سادگی حل می‌شود. برای این کار لازم است که به این نکته توجه کنیم که مقداری که شرایط گفته شده را فراهم کند مهم نیست و فقط کمینه کردن مقدار هزینه برای ما مهم است. برای این کار آرایه dp را می سازیم و مقدار هر خانه به ازای x را مشخص میکنیم. در ادامه با توجه به اینکه اگر مقدار کوچکتری از x با هزینه کمتر وجود داشته باشد، میتوان از این مقدار استفاده کرد، لازم است که هر خانه از dp را برابر با min خانه های قبل در نظر بگیریم. پس از مشخص شدن خانه های dp، کافی است که به ازای همه مقادیر bi، هزینه شامل انتقال آیتم bi (که برابر با i است) و هزینه انتقال یکی از aj ها به ابتدای a به نحوی که a کوچکتر از b شود،(که در dp ذخیره شده است) را بدست آوریم. min مقادیر بدست آمده جواب سوال است.

C++:

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

int main() {
    int tc;
    cin >> tc;
    while (tc--) {
        int n;
        cin >> n;
        int a[n], b[n];
        for (int i = 0; i < n; ++i) cin >> a[i];
        for (int i = 0; i < n; ++i) cin >> b[i];

        int dp[2 * n + 100];
        for (auto &t : dp) t = INT_MAX;

        for (int i = 0; i < n; ++i) {
           dp[a[i]] = i;
        }

        for (int i = 1; i < 2 * n + 10; ++i) {
```

```
            dp[i] = min(dp[i - 1], dp[i]);
        }

        int ans = INT_MAX;

        for (int i = 0; i < n; ++i) {
            ans = min(ans, dp[b[i]] + i);
        }
        cout << ans << endl;
    }
}
```