

راه حل سوال های کانتست (Div2) UICPC Round #2

سوال A: <mark>1311A</mark>

- 💠 اگر a و b **مساوی** باشند، آنگاه جواب ما **صفر** خواهد بود.
- ❖ اگر a > b باشد و a b عددی زوج باشد یا اینکه a < b و a < b عددی فرد باشد آنگاه جواب حتما 1 می شود.
 - 🌣 در غیر این صورت جوابی برابر با 2 خواهیم داشت.
 - 💠 در کد سوال مشاهده می شود که با عمل گرهای بیتی شرط ها پیاده سازی شده اند.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    cin >> t;
    while (t--)
    {
        int a, b;
        cin >> a >> b;
        if (a == b)
            cout << 0 << endl;</pre>
        else
             cout << 1 + int((a < b) ^ ((b - a) & 1)) << endl;</pre>
    }
    return 0;
}
```

<mark>Python:</mark>

```
for _ in range(int(input())):
    a, b = map(int, input().split())
    print(0 if a == b else 1+int((a < b) ^ ((b-a) & 1)))</pre>
```



♦ اگر n = 0 انگاه بازیکن بعدی بازی را می بازد

(هر چند در ابتدا با توجه به محدوده n این حالت امکان پذیر نیست).

- n زوج باشد، Mahmoud باید عددی زوج انتخاب کند و آن مساوی با n خواهد بود؛ در غیر این صورت (n)
 زوج) Mahmoud باید عددی کوچکتر از n انتخاب کند.
- حال اگر n عددی فرد و a عددی زوج باشد تفاضل آنها عددی فرد خواهد شد و از آنجایی که Ehab باید عددی
 فرد انتخاب کند ، عددی مساوی n انتخاب کرده و برنده خواهد شد.
- بنابراین اگر n زوج باشد Mahmoud برنده می شود و اگر n فرد باشد Ehab برنده میشود. اگر n = 1 باشد،
 آنگاه Mahmoud توانایی انتخاب عدد دیگری را نداشته و Ehab برنده میشود.

C++:

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

int main()
{
    int n;

    cin >> n;
    if (n % 2 == 0)
        cout << "Mahmoud" << endl;
    else
        cout << "Ehab" << endl;

    return 0;
}</pre>
```

```
print(['Mahmoud', 'Ehab'][int(input()) % 2])
```

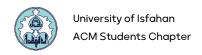


- ❖ اول از همه تعداد عدد های غیر تکراری آرایه a را d مینامیم.
- اکنون دنباله ای به تعداد d از آرایه α میسازیم و کوچکترین عنصر را از کپی اول، دومین عنصر را از کپی دوم بر
 میداریم و به همین ترتیب ادامه می دهیم تا کپی آخر.
 - ❖ به زبان ساده تر جواب ما در واقع تعداد عناصر غیر تکراری آرایه a میباشد.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    scanf("%d", &t);
    while (t--)
    {
        int n;
        scanf("%d", &n);
        set<int> s;
        while (n--)
        {
            int a;
            scanf("%d", &a);
            s.insert(a);
        printf("%d\n", s.size());
    }
    return 0;
}
```

```
t = int(input())
for _ in range(t):
    n = int(input())
    print(len(set(input().split())))
```

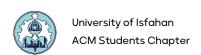


سوال D: <mark>1447B</mark>

- ♦ X را تعداد درایه های منفی ماتریس و S را مجموع قدر مطلق همه درایه ها در نظر می گیریم
- 💠 توجه کنید که این عملیات (ضرب در 1-) را میتوانیم برای هر دو درایه ای که به هم راه دارند انجام دهیم.
 - 🌣 اگر ماتریس ما دارای عدد **صفر** باشد آنگاه جواب مسئله همان S است. (چرا؟)
- در غیر اینصورت اگر X زوج باشد می توان همه آنها مثبت کرد و در اینصورت جواب ما همان S است . ولی اگر
 X فرد باشد و عدد صفر نیز موجود نباشد ، حداقل یک عدد منفی باقی می ماند پس بزرگترین عدد منفی را پیدا
 کرده و بقیه را مثبت میکنیم و در آخر آن عدد را با S جمع میکنیم.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main()
    int n;
    cin >> n;
    for (int i = 0; i < n; i++)
        int k, m, n;
        int ans = 0;
        int cnt = 0;
        int b = 100;
        cin >> m >> n;
        for (int j = 0; j < m * n; j++)
            cin >> k;
            ans += abs(k);
            if (k < 0)
                cnt++;
            b = min(abs(k), b);
        if (cnt % 2)
            ans -= 2 * b;
        cout << ans << endl;</pre>
    }
    return 0;
}
```



```
for _ in range(int(input())):
    n, m = map(int, input().split())
    a = []
    neg = 0

    for i in range(n):
        a += list(map(int, input().split()))

    neg = sum(map(lambda x: x < 0, a))

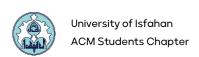
    b = [abs(x) for x in a]
    if neg % 2 == 0:
        print(sum(b))
    else:
        print(sum(b) - 2 * min(b))</pre>
```

سوال E: <mark>1251B</mark>

- میدانیم که طول رشته ها با این عملیات تغییری نمیکند.
- اگر طول رشته زوج باشد، درصورتی میتواند بازتابی شود که تعداد صفر ها و تعداد یک های آن رشته زوج باشد
 و درصورت فرد بودن طول رشته، همیشه میتوان آن را به رشته ای که بازتابی است تبدیل کرد.
- حال سوال این است که چگونه میتوان رشته ای به اصطلاح بد با طول زوج که تعداد صفر ها و یک های آن فرد است را به رشته ای قابل بازتابی شدن تبدیل کنیم؟
- برای این کار، اگر حداقل یک رشته به طول فرد داشته باشیم، میتوانیم از رشته ی بد به رشته با طول فرد یک بار
 عملیات را انجام دهیم. (جابهجایی یک با صفر یا صفر با یک) اگر رشته ای با طول فرد وجود نداشت هم
 میتوان با جابهجایی بین دو رشته ی بد هردو آنها را به رشته های قابل بازتایی شدن تبدیل کرد.
- در نتیجه اگر هم رشته ای به طول فرد نداشته باشیم و هم تعداد رشته های به طول زوجمان هم فرد تا
 باشند، همه ی رشته ها به جزیکی میتوانند به قابل بازتابی تبدیل شوند و در غیر این صورت، جواب تعداد کل
 رشته هاست.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main()
    int t;
    cin >> t;
    while (t--)
        int n;
        cin >> n;
        int cnt = 0, k = 0, ca = n;
        while (n--)
             string x;
             cin >> x;
             if (x.size() % 2)
                 k = 1;
             for (int i = 0; i < x.size(); i++)
                 if (x[i] == '0')
                      cnt++;
        if (cnt \% 2 == 0 || k)
             cout << ca << endl;</pre>
        else
             cout << ca - 1 << endl;</pre>
    }
}
```



```
for nt in range(int(input())):
    n = int(input())
    a = []
    for i in range(n):
        a.append(input())
    o = 0
    for i in a:
        o += i.count("0")
    flag = 0
    for i in a:
        if len(i) % 2 == 1:
            flag = 1
            break
    if (not flag) and o % 2:
        print(n-1)
    else:
        print(n)
```

سوال F: <mark>1315C</mark>

- این مساله با کمک الگوریتم greedy حل می شود.
- همانطور که گفته شده bi با مینیمم α2i-1, α2i برابر است. پس یکی از این دو عدد باید برابر با bi باشد.
 حال سوال این است که کدام یک؟
- العند مورت توضیح داد که جایگشتی از حروف است که با شرایط میتوان به این صورت توضیح داد که جایگشتی از حروف است که با شرایط موجود در مساله، زود تر از بقیه رشته ها ظاهر شود. یعنی اگر رشته از عدد تشکیل شده است عدد های کوچک تر،
 یا اگر حروف تشکیل شده حروفی که در الفبا اول تر هستند، زود تر ظاهر شوند. پس از بین a2i و a2i باید a2i بایر با bi در نظر بگیریم تا عدد کوچک تر قبل از عدد بزرگ تر آمده باشد.
 - ❖ اکنون که میدانیم a1=b1, a3=b2, a5=b3 و... به جای اندیس های زوج a باید چه عدد هایی را قرار دهیم؟
- بیس میتوانیم a2>a1 ریرا (a2!=a1 and b1 = min(a1.a2)). پس میتوانیم a2 را برابر با مینیمم x ای در نظر بگیریم که میدانیم x ای وجود نداشته باشد x و x ایرا این ها را از قبل پر کرده ایم. اگر همچین x ای وجود نداشته باشد (یعنی همه اعداد بزرک تر از a1 استفاده شده باشند) جوابی برای مساله وجود ندارد.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main()
    int t;
    cin >> t;
    while (t--)
        int n, i, j;
        cin >> n;
        int b[2 * n];
        bool visited[2 * n + 1] = {0};
        for (i = 0; i < 2 * n; i += 2)
            cin >> b[i];
            visited[b[i]] = 1;
        for (i = 1; i < 2 * n; i += 2)
        {
            j = b[i - 1] + 1;
            while (j \le 2 * n \&\& visited[j] == 1)
                 j++;
            if (j > 2 * n)
                 break;
            visited[j] = 1;
            b[i] = j;
        }
        if (i < 2 * n)
            cout << -1 << endl;</pre>
        else
        {
            for (i = 0; i < 2 * n; i++)
                cout << b[i] << " ";</pre>
            cout << endl;</pre>
        }
    }
    return 0;
}
```

```
for _ in range(int(input())):
    n = int(input())
    b = list(map(int, input().split()))
    ans = []
    for i in b:
        ans.append(i)
        t = i+1
        while(t in b) or (t in ans):
            t += 1
        ans.append(t)
    if(max(ans) == 2*n):
        print(*ans)
    else:
        print("-1")
```