美丽数

- 考点：**贪心**、分类讨论
- 难度 CSP-J T2

测试点 1~2

给爆搜的部分分。

测试点5~6

特殊部分分。

当 a 中只有两个数的时候，设分别是 x, y ，那么最终的美丽数只有可能是 $xyxy\dots xyx$ 、 $xyxy\dots xy$ 、$yxyx\dots yx$、$yxyx\dots yxy$ 其中之一。

分类讨论一下即可。

测试点 7~8

a 中只有三个数非 0 ，设分别是 x, y, z 。

贪心的考虑，我们从数位高到低来考虑，那么我们肯定会先放最小的（如果前一个位放了最小的那就放次小的）。

但什么时候不能这样放呢？就是剩下的 x, y, z 不能形成一个美丽数了。

考虑一下就能知道 x, y, z 能形成美丽数的充要条件就是 $a_x \leq a_y + a_z + 1$，我们设 $a_x > a_y > a_z$ 。

所以判断一下就可以了。

ps：如何处理不能有前导 0 ？ 可以设前一个放的数是 0 。

测试点 3~4

可以沿用上一档部分分的做法，从数位高到低考虑，对每一位依次考虑 $0 \sim 9$ 如果放在这个位了之后，剩下的数还能不能形成一个美丽数。

由于 n 不大，所以我们可以暴力的去判断，复杂度为 $O(n^2)$。

测试点 9~10

关键在于如何判断当前的数还能不能形成一个美丽数。

我们设 $mx = \max\{a_0, a_1, \dots, a_9\}$，$su = \sum_{i=0}^{9} a_i$ ，那么能形成的充要条件就是 $mx \leq su - mx + 1$ 。

具体写法可以参考std。

ps：你有没有考虑输出为 0 的情况？虽然数据里是没有的（不能太刁钻了），但还是要注意的。

T2

当相邻两个人的身高排名不是连续的，那么最终一定会在这两个人中间分割一次队列。

反过来推，如果所有队列中的人身高排名连续，那么一定可以通过组合队列达成需求。

因此答案就是不连续的身高排名个数，或是连续子序列数-1。

```cpp
#include<bits/stdc++.h>
using namespace std;
int n;
int a[1000005];
int ans=0;
int up=2;
int main(){
    scanf("%d",&n);
    a[0]=-114514;
    for(int i=1;i<=n;i++){
        scanf("%d",&a[i]);
        if((up==2 || up==1) && a[i]==a[i-1]+1) up=1;
        else if((up==2 || up==0) && a[i]==a[i-1]-1) up=0;
        else{
            ans++;
            up=2;
        }
    }
    cout<<ans-1;
    return 0;
}
```

维护集合

- 考点：**桶排序、初等数论**
- 难度 CSP T2+

测试点1~10

留给暴力的做法。

就是暴力对于每一次询问 x，暴力考虑集合中的数 y，求出 y 是 x 的几重约数。

复杂度是 $O(qn)$

测试点 11~12

虽然 n 很大，但是 a_i, t 很小，也就是如果我们能够去除重复的数字，就可以快速的求出答案。

所以我们就可以用桶来维护第 i 个数字出现了几次。

测试点13~14

虽然 a_i, t 很大，但是 x 很小，也就是有效的 a_i, t 很小，所以我们只需要在每次查询的时候考虑前 300 个数就可以了。

维护数字是否出现和测试点11~12的做法一样，还是用桶来维护。

测试点15~16

在没有修改的情况下，我们可以提前把所有答案处理出来。

测试点 17~20

算是比较有难度的一档分，难点在于如何快速求一个数的 k 重约数都是谁。

其实我只要对 x 的每个约数来考虑一下他到底是几重约数。

如何找到 x 的所有约数，我们可以枚举 $1 \sim \sqrt{x}$ ，若 i 是 x 的约数，那么 $\frac{x}{i}$ 也是 x 的约数，并且一定是 1 重约数。

时间复杂度为 $O(q\sqrt{t})$

T4

$60pts$

对大臣按照最高需求排序，然后按价值从小到大贪心查找金砖能匹配的大臣即可，时间复杂度$O(n^2)$。

$100pts$

将大臣的最低需求从小到大排序，然后从小到大枚举价值，如果价值达到了某个大臣的最低工资，那么存储大臣的最高工资。然后再尝试将该价值的金砖与最高工资的最小值匹配即可。大臣的最高工资可用集合或优先队列维护，时间复杂度为$O(nlogn)$

```cpp
#include<bits/stdc++.h>
using namespace std;
int c,l;
pair<int,int> person[1000005];
int money[1000005];
int ans=0;
int point=1;
multiset<int> st;
int main() {
    scanf("%d%d",&c,&l);
    for(int i=1; i<=c; i++)
scanf("%d%d",&person[i].first,&person[i].second);
    sort(person+1,person+1+c);
    while(l--){
        int a;
        scanf("%d",&a);
        money[a]+=1;
    }
    for(int i=1;i<=1000;i++){
        while(person[point].first<=i && point<=c)
st.insert(person[point++].second);
        while(st.size() && *st.begin()<i) st.erase(st.begin());
        for(int j=1;j<=money[i];j++){
            if(st.empty()) break;
            st.erase(st.begin());
            ans++;
        }
    }
    cout<<ans;
    return 0;
```

```
29 }
```