

中位数问题

acwing 104 货仓选址

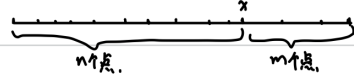
思路

取坐标的中位数，作为仓库的位置。

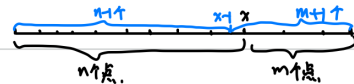
证明



① 不妨令货仓在位置 x ，且有 x 左侧点的数量 n 大于右侧 m 。



② 若货仓位置左移一位。

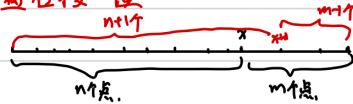


总距离 $dis = n + m$

而 $n > m$

有 $dis - n + m < dis$ ，距离变小

③ 若货仓位置右移一位



总距离 $dis = m + n$

而 $n < m$

有 dis 现在 $> dis$ 原来，距离变大

30 / 30

代码

```
1 #include <bits/stdc++.h>
2 #define endl "\n"
3
4 using namespace std;
5
6 const int maxn = 100010;
7 int a[maxn];
8 int res;
9
10 int main(){
11     ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
12     int n; cin >> n;
13     for(int i = 1; i <= n; i++){
14         cin >> a[i];
15     }
16
17     sort(a + 1, a + 1 + n);
```

```

18
19     int mid = 1 + n >> 1;
20
21     for(int i = 1; i <= n; i++){
22         res += abs(a[mid] - a[i]);
23     }
24
25     cout << res << endl;
26
27     return 0;
28 }

```

acwing 1536 均分纸牌

思路

如果有解 -> 总牌数能被整除

从最左边开始

若第一堆牌的数量不足平均数 -> 向第二堆借相应的牌数,直到平均数

若第一堆牌的数量大于平均数 -> 给第二堆牌相应的牌数,直到平均数

...

以此类推,直到最后一堆牌,其数量也一定是平均数

代码

```

1  #include <bits/stdc++.h>
2  #define endl "\n"
3
4  using namespace std;
5
6  const int maxn = 110;
7  int n, sum, ans;
8  int a[maxn];
9
10 int main(){
11     ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
12     cin >> n;
13
14     for(int i = 1; i <= n; i++){
15         cin >> a[i];
16         sum += a[i];
17     }
18
19     int avg = sum / n;
20
21     for(int i = 1; i <= n; i++){
22         if(a[i] != avg) a[i + 1] += a[i] - avg, ans ++;
23     }
24
25     cout << ans << endl;
26     return 0;
27 }

```

[acwing 1208 翻硬币 \(非中位数问题,但与上题贪心思想类似\)](#)

思路

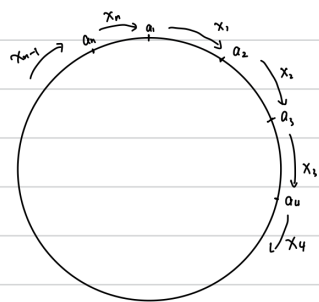
对一个字符串，从前向后遍历。如有不一致，则将该位置与该位置之后以为的字符反转。

代码

```
1  #include <bits/stdc++.h>
2  #define endl "\n"
3
4  using namespace std;
5
6  string s1, s2;
7  int cnt;
8
9  char change(char s){
10     if(s == 'o') return '*';
11     else return 'o';
12 }
13
14 int main(){
15     ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
16
17     cin >> s1 >> s2;
18
19     for(int i = 0; i < s1.length(); i++){
20         if(s1[i] != s2[i]){
21             cnt++;
22             s1[i] = change(s1[i]);
23             s1[i + 1] = change(s1[i + 1]);
24         }
25     }
26
27     cout << cnt;
28
29     return 0;
30 }
```

[acwing 122 糖果传递](#)

思路 + 证明



$$b = \frac{1}{n} (a_1 + a_2 + \dots + a_n)$$

目标: 最小化 $|x_1| + |x_2| + |x_3| + \dots + |x_n|$

$$\begin{cases} a_1 - x_1 + x_n = b \\ a_2 - x_2 + x_1 = b \\ \vdots \\ a_n - x_n + x_{n-1} = b \end{cases} \Rightarrow \begin{cases} x_1 = x_n - (b - a_1) \\ x_2 = x_1 - (b - a_2) = x_n - (2b - a_1 - a_2) \\ \vdots \\ x_n = x_n - (nb - a_1 - a_2 - \dots - a_{n-1}) \end{cases}$$

$$\text{原式} = |x_n - (nb - a_1)| + |x_n - (2b - a_1 - a_2)| + \dots + |x_n - a_1|$$

$$= |x - c_1| + |x - c_2| + |x - c_3| + \dots + |x - c_{n-1}|$$

转化为货仓选址.

$\Rightarrow x$ 取中位数.

代码

```

1  #include <bits/stdc++.h>
2  #define endl "\n"
3  typedef long long ll;
4  using namespace std;
5  const int maxn = 1e6 + 10;
6
7  ll a[maxn], s[maxn], c[maxn];
8  int n, ans;
9  ll sum;
10
11 int main(){
12     ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
13     cin >> n;
14     for(int i = 1; i <= n; i++){
15         cin >> a[i];
16         s[i] = s[i - 1] + a[i];
17     }
18
19     ll avg = s[n] / n;
20
21     for(int i = 1; i <= n; i++){
22         c[i - 1] = i * avg - s[i];
23     }
24
25
26     nth_element(c, c + n / 2, c + n);
27     ll res = 0;
28     for(int i = 0; i < n; i++){
29         res += abs(c[i] - c[n / 2]);
30     }
31
32     cout << res;
33     return 0;
34 }
35

```

acwing 105 七夕祭

思路

对于每一行的元素，我们可以求出他们的行和。而在行和中首尾元素相连，这就转化为糖果传递问题。

又因为，我们一次移动两个数，仅仅会影响其单一行或单一列的目标，而不会同时影响一行和一列，所以我们单独处理两个。如果合法，那么就两个答案加起来，如果单个合法，那么就处理单个就好了。

代码

```
1  #include <bits/stdc++.h>
2  #define endl "\n"
3  typedef long long ll;
4
5  using namespace std;
6
7  const int maxn = 1e5 + 10;
8  int n, m;
9  int row[maxn], col[maxn], s[maxn], c[maxn];
10
11 ll work(int a[], int n){ //环形均分糖果问题
12     for(int i = 1; i <= n; i++){
13         s[i] = s[i - 1] + a[i];
14     }
15
16     if(s[n] % n) return -1;
17
18     int avg = s[n] / n;
19
20     c[1] = 0;
21     for(int i = 2; i <= n; i++){
22         c[i] = s[i - 1] - (i - 1) * avg;
23     }
24
25     sort(c + 1, c + n + 1);
26     ll res = 0;
27     for(int i = 1; i <= n; i++) res += abs(c[i] - c[(n + 1) / 2]);
28
29     return res;
30 }
31
32 int main(){
33     ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
34
35     int t; cin >> n >> m >> t;
36
37     while(t--){
38         int x, y; cin >> x >> y;
39         row[x] ++, col[y] ++;
40     }
41
42     ll r = work(row, n);
43     ll c = work(col, m);
44 }
```

```
45     if(r != -1 && c != -1) cout << "both " << r + c;  
46     else if(r != -1) cout << "row " << r;  
47     else if(c != -1) cout << "column " << c;  
48     else cout << "impossible";  
49  
50     return 0;  
51 }
```

[acwing 106 动态中位数](#)

思路

代码