

第五届河南省CCPC河南省省赛题解+复盘

A - 小水獭游河南 — 签到

这个题关键点就是 知道a串最多有26个字母，超过26个字母一定会重复

同时注意，一旦发生重复后边也一定会重复，记得break

还要要特判一下s长度为1的情况

```
1  #include <iostream>
2  #include <cstring>
3  #include <algorithm>
4  #include <vector>
5  using namespace std;
6
7  // 判断回文
8  bool check(string s, int st)
9  {
10     for(int i = st, j = s.size() - 1; i < j; i ++, j --)
11         if(s[i] != s[j]) return false;
12     return true;
13 }
14
15 void solve()
16 {
17     string s; cin >> s;
18     vector<int> cnt(26, 0);
19
20     if(s.size() == 1) {
21         cout << "NaN\n";
22         return ;
23     }
24
25     // 最多26个字母，st代表后边回文字符串的开始位置
26     for(int st = 1; st <= 26; st ++){
27         // 之前的计数
```

```

28
29     int t = s[st - 1] - 'a';
30     // 判断是否重复, 重复直接break
31     if(cnt[t]) {
32         break;
33     }
34     cnt[t] ++;
35
36     // 检查是否回文
37     if(check(s, st))
38     {
39         cout << "HE\n";
40         return;
41     }
42 }
43 cout << "NaN\n";
44 }
45
46 int main()
47 {
48     int T; cin >> T;
49     while(T --) solve();
50 }
51

```

B - Art for Rest – 找性质+前缀/后缀处理

原题意是 把一个长度为 n 的数组, 分为每段长度为 k 的区间 (最后一个区间可能不足 k) , 每个区间单独排序, 区间排序后拼接形成整个数组。求满足拼接形成的数组是非严格递增的 k 的 数量

题目可转化为 求符合 每一段长度为 k 的区间的最大值 小于等于 后缀数组 所有数的最小值的 k 的取值数量

又可以转化为 求符合 每个区间最后一个位点 前缀最大值 小于等于 后缀最小值的 k 的取值数量

直接暴力即可, 调和级数复杂度 $n\log n$

```

1  #include <iostream>
2  #include <cstring>
3  #include <algorithm>
4  #include <vector>

```

```

5  using namespace std;
6
7  int main()
8  {
9      int n; cin >> n;
10     vector<int> a(n + 1), minv(n + 2), st(n + 2, 0);
11
12     for(int i = 1; i <= n; i++) scanf("%d", &a[i]);
13
14     // 求后缀最小值
15     minv[n + 1] = 2e9;
16     for(int i = n; i >= 1; i--)
17         minv[i] = min(minv[i + 1], a[i]);
18
19     // st[i] = 1 表示前i个数里的最大值小于 后缀所有数的最小值
20     int maxv = 0;
21     for(int i = 1; i <= n; i++)
22     {
23         maxv = max(maxv, a[i]);
24         if(maxv <= minv[i + 1] || i == n)
25             st[i] = 1;
26     }
27
28
29     int res = 0;
30     // 对于每一个长度为k的区间，看最后一个位置 st[i] 全部等于1 即满足条件，答案 + 1
31     for(int k = 1; k <= n; k++)
32     {
33         int f = 1;
34         for(int i = k; i <= n; i += k)
35         {
36             if(st[i] == 0)
37             {
38                 f = 0;
39                 break;
40             }
41         }
42         res += f;
43     }

```

```
44     printf("%d\n", res);
45 }
```

C - Toxel 与随机数生成器 — 思维题

因为 $|s| = 1e6$ 每段长度 $1e3$ 到 $1e4$ ，直接暴力如果重复出现100次以上就判为No

```
1  #include <iostream>
2  #include <cstring>
3  #include <algorithm>
4  #include <vector>
5  using namespace std;
6
7  int main()
8  {
9      string s; cin >> s;
10     // 记录重复数量
11     int cnt = 0;
12     string tar = s.substr(0, 1000);
13     for(int i = 0; i + 1000 < s.size(); i++)
14     {
15         if(tar == s.substr(i, 1000)){
16             cnt++;
17             i += 1000;
18         }
19     }
20     if(cnt < 100) cout << "Yes\n";
21     else cout << "No\n";
22 }
```

E - 矩阵游戏 — dp

官方题解说的很清楚，不过需要注意的是三维转二维的过程，很容易出现错误，用一个新的数组记录上一个状态

考虑 $f_{i,j,k}$ 表示从 $(1,1)$ 开始走到 (i,j) 恰好替换了 k 个？最多获得的分数，容易得到转移方程：

$$f_{i,j,k} = \begin{cases} \max(f_{i,j-1,k}, f_{i-1,j,k}) & s_{i,j} = 0 \\ \max(f_{i,j-1,k}, f_{i-1,j,k}) + 1 & s_{i,j} = 1 \\ \max(f_{i,j-1,k}, f_{i-1,j,k}, f_{i-1,j,k-1} + 1, f_{i,j-1,k-1} + 1) & k \neq 0 \wedge s_{i,j} = ? \\ \max(f_{i,j-1,k}, f_{i-1,j,k}) & k = 0 \wedge s_{i,j} = ? \end{cases}$$

CSDN @_WAWA鱼_

```
1  #include <iostream>
2  #include <algorithm>
3  #include <cstring>
4  #include <vector>
5  using namespace std;
6
7  void solve()
8  {
9      int n, m, x; cin >> n >> m >> x;
10     vector<vector<int>> f(m + 1, vector<int>(x + 1)), g(f);
11     vector<string> s(n + 1);
12     for (int i = 1; i <= n; i++) cin >> s[i], s[i] = " " + s[i];
13
14     int res = 0;
15     // g 对应的其实就是 f[i-1][j][k] f对应 f[i][j][k]
16     for (int i = 1; i <= n; i++)
17     {
18         for (int j = 1; j <= m; j++)
19         {
20             for (int k = 0; k <= x; k++)
```

```

21         {
22             if (s[i][j] == '0') f[j][k] = max(f[j - 1][k], g[j][k]);
23             else if (s[i][j] == '1') f[j][k] = max(f[j - 1][k], g[j][k]) + 1;
24             else
25             {
26                 if (k >= 1) f[j][k] = max(f[j - 1][k - 1] + 1, g[j][k - 1] + 1);
27                 else f[j][k] = max(f[j - 1][k], g[j][k]);
28             }
29             if (i == n && j == m) res = max(res, f[j][k]);
30         }
31     }
32     g = f;
33 }
34
35 cout << res << "\n";
36 }
37
38 int main()
39 {
40     int T; cin >> T;
41     while(T --) solve();
42 }
43

```

F - Art for Last — 贪心 + 区间求最小值（下边三种写法均可）

题意是求 从n个数中 选k个数 求k个数中（任意2个数之差的最小值）乘 （任意2数的之差最大值） 的最小值

我们要想求最小值，就是尽量让2个数之差的最小值尽可能的小，让任意2数的之差最大值也尽可能的小

直接贪心排序一下，最大值就是长度为k的区间 最后一个-第一个。

然后就是找区间中两个数差值的最小值，差值的最小值一定在排序后相邻的元素中产生

因此找相邻元素差值最小值即可，把相邻元素差值单独取出来，用ST表或线段树或滑动窗口维护都可以，求一下区间最小值即可

所以sort后遍历一下求每个长度k区间的最大最小值，取乘积最小即可

写法一 ST表写法

```

1  #include <iostream>
2  #include <algorithm>

```

```

3  #include <cstring>
4  #include <vector>
5  using namespace std;
6  #define int long long
7  const int N = 500010;
8  int Fmin[N][20];
9  int lg2[N];
10 int a[N];
11 int n, k;
12 void init_log()
13 {
14     lg2[0] = -1;
15     for(int i = 1; i < N; i++) lg2[i] = lg2[i >> 1] + 1;
16 }
17 void init()
18 {
19     for(int i = 1; i < n; i++) Fmin[i][0] = a[i + 1] - a[i];
20     int k = lg2[n];
21     for(int j = 1; j <= k; j++)
22         for(int i = 1; i <= n - (1 << j) + 1; i++)
23             {
24                 Fmin[i][j] = min(Fmin[i][j - 1], Fmin[i + (1 << j - 1)][j - 1]);
25             }
26 }
27 }
28 int RMQ(int l,int r)
29 {
30     int k = lg2[r - l + 1];
31     int minv = min(Fmin[l][k], Fmin[r - (1 << k) + 1][k]);
32     return minv;
33 }
34
35 signed main()
36 {
37     init_log();
38
39     scanf("%lld %lld",&n, &k);
40     for(int i = 1; i <= n; i++) scanf("%lld", a + i);
41     sort(a + 1, a + n + 1);

```

```

42
43     init();
44     int res = 2e18;
45     for(int i = k; i <= n; i ++)
46     {
47         int t = RMQ(i - k + 1, i - 1) * (a[i] - a[i - k + 1]);
48         res = min(t, res);
49     }
50     cout << res << '\n';
51 }
52

```

写法二 滑动窗口

```

1  #include <iostream>
2  #include <cstring>
3  #include <algorithm>
4  #define int long long
5  using namespace std;
6
7  signed main()
8  {
9      int n, k; cin >> n >> k;
10     vector<int> a(n + 1), d(n + 1), q(n + 1);
11     for(int i = 1; i <= n; i ++) cin >> a[i];
12     sort(a.begin() + 1, a.end());
13
14     for(int i = 1; i < n; i ++)
15         d[i] = a[i + 1] - a[i];
16
17     int hh = 0, tt = -1;
18     int res = 2e18;
19     for(int i = 1; i <= n; i ++)
20     {
21         if(hh <= tt && q[hh] < i - k + 1) hh ++;
22         if(i >= k)
23         {
24             int t = d[q[hh]] * (a[i] - a[i - k + 1]);
25             res = min(res, t);
26         }
27     }
28     cout << res << '\n';
29 }

```



```

25
26     }
27     while(hh <= tt && d[i] <= d[q[tt]]) tt --;
28     q[++ tt] = i;
29 }
30 cout << res << '\n';
31 }

```

写法三 multiset写法

```

1  #include <iostream>
2  #include <cstring>
3  #include <algorithm>
4  #include <set>
5  #define int long long
6  using namespace std;
7
8  signed main()
9  {
10     ios::sync_with_stdio(0);
11     cin.tie(0);cout.tie(0);
12     int n, k; cin >> n >> k;
13     vector<int> a(n + 1);
14     for(int i = 1; i <= n; i++) cin >> a[i];
15     sort(a.begin() + 1, a.end());
16
17     multiset<int> S;
18     for(int i = 1; i < k; i++)
19         S.insert(a[i + 1] - a[i]);
20
21     int res = *S.begin() * (a[k] - a[1]);
22     for(int i = k; i <= n; i++)
23     {
24         int t = *S.begin() * (a[i] - a[i - k + 1]);
25         res = min(res, t);
26         S.erase(S.find(a[i - k + 2] - a[i - k + 1]));
27         S.insert(a[i + 1] - a[i]);
28     }
29     cout << res << '\n';

```

```
29
30 }
```

G - Toxel 与字符画 — 大模拟

大模拟，容易出错的地方是判断结果是否超过 $1e18$ 输出INF

这里不需要用什么快速幂，特判一下如果 $x = 1$ 结果为1

否则只要 x 是比2大的数，2的60次方超过 $1e18$ ，因此遍历不会超过60次

直接暴力判断即可，需要注意暴力乘的时候可能爆longlong，直接用__int128就好了

```
1  #include <iostream>
2  #include <algorithm>
3  #include <cstring>
4  #include <vector>
5  using namespace std;
6  #define int long long
7  string big[11][10], sma[11][10];
8  string dengyu[10], inf[40];
9  char g[15][2010];
10 int col = 0;
11 void init()
12 {
13     big[0][0]=".....",sma[0][0]=".....";
14     big[0][1]=".....",sma[0][1]=".00000";
15     big[0][2]=".0000000",sma[0][2]=".0...0";
16     big[0][3]=".0.....0",sma[0][3]=".0...0";
17     big[0][4]=".0.....0",sma[0][4]=".0...0";
18     big[0][5]=".0.....0",sma[0][5]=".00000";
19     big[0][6]=".0.....0",sma[0][6]=".....";
20     big[0][7]=".0.....0",sma[0][7]=".....";
21     big[0][8]=".0000000",sma[0][8]=".....";
22     big[0][9]=".....",sma[0][9]=".....";
23
24     big[1][0]=".....",sma[1][0]=".....";
25     big[1][1]=".....",sma[1][1]=".....1";
26     big[1][2]=".....1",sma[1][2]=".....1";
27     big[1][3]=".....1",sma[1][3]=".....1";
28     big[1][4]=".....1",sma[1][4]=".....1";
```

```

29     big[1][5]=".....1",sma[1][5]=".....1";
30     big[1][6]=".....1",sma[1][6]=".....";
31     big[1][7]=".....1",sma[1][7]=".....";
32     big[1][8]=".....1",sma[1][8]=".....";
33     big[1][9]=".....",sma[1][9]=".....";
34
35     big[2][0]=".....",sma[2][0]=".....";
36     big[2][1]=".....",sma[2][1]=".22222";
37     big[2][2]=".2222222",sma[2][2]=".....2";
38     big[2][3]=".....2",sma[2][3]=".22222";
39     big[2][4]=".....2",sma[2][4]=".2....";
40     big[2][5]=".2222222",sma[2][5]=".22222";
41     big[2][6]=".2.....",sma[2][6]=".....";
42     big[2][7]=".2.....",sma[2][7]=".....";
43     big[2][8]=".2222222",sma[2][8]=".....";
44     big[2][9]=".....",sma[2][9]=".....";
45
46     big[3][0]=".....",sma[3][0]=".....";
47     big[3][1]=".....",sma[3][1]=".33333";
48     big[3][2]=".3333333",sma[3][2]=".....3";
49     big[3][3]=".....3",sma[3][3]=".33333";
50     big[3][4]=".....3",sma[3][4]=".....3";
51     big[3][5]=".3333333",sma[3][5]=".33333";
52     big[3][6]=".....3",sma[3][6]=".....";
53     big[3][7]=".....3",sma[3][7]=".....";
54     big[3][8]=".3333333",sma[3][8]=".....";
55     big[3][9]=".....",sma[3][9]=".....";
56
57     big[4][0]=".....",sma[4][0]=".....";
58     big[4][1]=".....",sma[4][1]=".4...4";
59     big[4][2]=".4.....4",sma[4][2]=".4...4";
60     big[4][3]=".4.....4",sma[4][3]=".44444";
61     big[4][4]=".4.....4",sma[4][4]=".....4";
62     big[4][5]=".4444444",sma[4][5]=".....4";
63     big[4][6]=".....4",sma[4][6]=".....";
64     big[4][7]=".....4",sma[4][7]=".....";
65     big[4][8]=".....4",sma[4][8]=".....";
66     big[4][9]=".....",sma[4][9]=".....";
67
    
```

```

68     big[5][0]=".....",sma[5][0]=".....";
69     big[5][1]=".....",sma[5][1]=".55555";
70     big[5][2]=".5555555",sma[5][2]=".5....";
71     big[5][3]=".5.....",sma[5][3]=".55555";
72     big[5][4]=".5.....",sma[5][4]=".....5";
73     big[5][5]=".5555555",sma[5][5]=".55555";
74     big[5][6]=".....5",sma[5][6]=".....";
75     big[5][7]=".....5",sma[5][7]=".....";
76     big[5][8]=".5555555",sma[5][8]=".....";
77     big[5][9]=".....",sma[5][9]=".....";
78
79     big[6][0]=".....",sma[6][0]=".....";
80     big[6][1]=".....",sma[6][1]=".66666";
81     big[6][2]=".6666666",sma[6][2]=".6....";
82     big[6][3]=".6.....",sma[6][3]=".66666";
83     big[6][4]=".6.....",sma[6][4]=".6...6";
84     big[6][5]=".6666666",sma[6][5]=".66666";
85     big[6][6]=".6.....6",sma[6][6]=".....";
86     big[6][7]=".6.....6",sma[6][7]=".....";
87     big[6][8]=".6666666",sma[6][8]=".....";
88     big[6][9]=".....",sma[6][9]=".....";
89
90     big[7][0]=".....",sma[7][0]=".....";
91     big[7][1]=".....",sma[7][1]=".77777";
92     big[7][2]=".7777777",sma[7][2]=".....7";
93     big[7][3]=".....7",sma[7][3]=".....7";
94     big[7][4]=".....7",sma[7][4]=".....7";
95     big[7][5]=".....7",sma[7][5]=".....7";
96     big[7][6]=".....7",sma[7][6]=".....";
97     big[7][7]=".....7",sma[7][7]=".....";
98     big[7][8]=".....7",sma[7][8]=".....";
99     big[7][9]=".....",sma[7][9]=".....";
100
101
102     big[8][0]=".....",sma[8][0]=".....";
103     big[8][1]=".....",sma[8][1]=".88888";
104     big[8][2]=".8888888",sma[8][2]=".8...8";
105     big[8][3]=".8.....8",sma[8][3]=".88888";
106     big[8][4]=".8.....8",sma[8][4]=".8...8";

```

```

107     big[8][5]=".8888888",sma[8][5]=".88888";
108     big[8][6]=".8.....8",sma[8][6]=".....";
109     big[8][7]=".8.....8",sma[8][7]=".....";
110     big[8][8]=".8888888",sma[8][8]=".....";
111     big[8][9]=".....",sma[8][9]=".....";
112
113     big[9][0]=".....",sma[9][0]=".....";
114     big[9][1]=".....",sma[9][1]=".99999";
115     big[9][2]=".9999999",sma[9][2]=".9...9";
116     big[9][3]=".9.....9",sma[9][3]=".99999";
117     big[9][4]=".9.....9",sma[9][4]=".....9";
118     big[9][5]=".9999999",sma[9][5]=".99999";
119     big[9][6]=".....9",sma[9][6]=".....";
120     big[9][7]=".....9",sma[9][7]=".....";
121     big[9][8]=".9999999",sma[9][8]=".....";
122     big[9][9]=".....",sma[9][9]=".....";
123
124     dengyu[0]=".....";
125     dengyu[1]=".....";
126     dengyu[2]=".....";
127     dengyu[3]=".....";
128     dengyu[4]=".=====";
129     dengyu[5]=".....";
130     dengyu[6]=".=====";
131     dengyu[7]=".....";
132     dengyu[8]=".....";
133     dengyu[9]=".....";
134
135     inf[0]=".....";
136     inf[1]=".....";
137     inf[2]=".IIIIIII.N....N.FFFFFFFF";
138     inf[3]="....I....NN...N.F.....";
139     inf[4]="....I....N.N...N.F.....";
140     inf[5]="....I....N..N..N.FFFFFFFF";
141     inf[6]="....I....N...N.N.F.....";
142     inf[7]="....I....N....NN.F.....";
143     inf[8]=".IIIIIII.N....N.F.....";
144     inf[9]=".....";
145
    
```

```

146 }
147
148 // 判断是否超过1e18
149 int check(int x, int y)
150 {
151     if(x == 1) return 1;
152
153     __int128 res = 1;
154     for(int i = 1; i <= y; i ++)
155     {
156         res *= x;
157         if(res > 1e18) return 0;
158     }
159     return res;
160 }
161
162 // 把数字提取到vector数组中
163 void tiq(int x, vector<int>& vec)
164 {
165     while(x)
166     {
167         vec.push_back(x % 10);
168         x /= 10;
169     }
170     reverse(vec.begin(),vec.end());
171 }
172
173 // 添加大数
174 void add_big(vector<int>& vec)
175 {
176     for(auto x : vec)
177     {
178         for(int i = 0; i < 10; i ++)
179             for(int j = 0; j < 8; j ++)
180                 g[i][col + j] = big[x][i][j];
181         col += 8;
182     }
183 }
184

```

```

185 // 添加小数
186 void add_small(vector<int>& vec)
187 {
188     for(auto x : vec)
189     {
190         for(int i = 0; i < 10; i++)
191             for(int j = 0; j < 6; j++)
192                 g[i][col + j] = sma[x][i][j];
193         col += 6;
194     }
195 }
196
197 // 添加等于号
198 void add_dengyu()
199 {
200     for(int i = 0; i < 10; i++)
201         for(int j = 0; j < 8; j++)
202             g[i][col + j] = dengyu[i][j];
203     col += 8;
204 }
205
206 // 添加INF
207 void add_inf()
208 {
209     for(int i = 0; i < 10; i++)
210         for(int j = 0; j < 24; j++)
211             g[i][col + j] = inf[i][j];
212     col += 24;
213 }
214
215 // 最后一列"."
216 void add_col()
217 {
218     for(int i = 0; i < 10; i++)
219         g[i][col] = '.';
220     col += 1;
221 }
222
223 // 打印结果

```

```

224 void print()
225 {
226     for(int i = 0; i < 10; i ++){
227         {
228             for(int j = 0; j < col; j ++){
229                 cout << g[i][j];
230                 cout << endl;
231             }
232         }
233
234 void solve()
235 {
236     col = 0;
237     int x, y;
238     scanf("%lld^{%lld}", &x, &y);
239
240     vector<int> vecx;
241     tiqu(x, vecx);
242     add_big(vecx);
243
244     vector<int> vecy;
245     tiqu(y, vecy);
246     add_small(vecy);
247
248     add_dengyu();
249
250     int res = check(x, y);
251     if(res == 0){
252         add_inf();
253     }
254     else{
255         vector<int> vecres;
256         tiqu(res, vecres);
257         add_big(vecres);
258     }
259     add_col();
260     print();
261 }
262

```



```
263 signed main()
264 {
265     init();
266     int T; cin >> T;
267     while(T --) solve();
268 }
269
```

H - Travel Begins — 贪心+推导

题意是 给一个数 n ，任意构造 k 个实数（可为0），他们的和为 n ，实数小数部分小于0.5的舍掉，大于等于0.5进位，使其 k 个数都为整数，求这样做后最小之和 和 最大之和

如果 $k > 2 * n$ 可确保 将每个数分为 $n/k < 1/2$ 最小为0， 或者每个数分为0.5,其余为0，最大值为 $2 * n$

如果 $k \leq 2 * n$ 直接贪心即可

最小的时候一定是尽可能让前 $k-1$ 个数为0.499999999...，这样前 $k-1$ 个数就会全部把小数舍掉

最大的时候一定是尽可能让前 $k-1$ 个数为0.5， 这样前 $k-1$ 个数全都变成1

然后让 n 减去前 $k-1$ 个数为第 k 个数，第 k 个数小数部分大于0.5就进1，最终求得结果

最小的时候

第 k 个数为 $n - (0.5 - eps) * (k - 1)$ 其中 eps 趋近于0

前 $k-1$ 个数之和为0，只需要判断第 k 个数是否进位即可

第 k 个数结果推导

```
res = n - (0.5 - eps) * (k - 1)    其中eps->0  
= n - (k - 1) / 2 + eps * (k - 1)  其中eps->0  
= n - (k - 1) / 2 + eps            其中eps->0
```

if $k - 1$ 为奇数

```
res = n - (k - 1) / 2
```

其中 $(k - 1) / 2$ 向下取整，少减了0.5，
但是即使减去后res - 1但小数会剩余 0.5 + eps会进位又 + 1，
所以减不减0.5无所谓

if $k - 1$ 为偶数 |

```
res = n - (k - 1) / 2;
```

故最小值为 $n - k / 2$;

CSDN @_WAWA鱼_

最大的时候

第k个数为 $n - 0.5 * (k - 1)$

前k-1个数之和为k-1

推导

因为前 $k-1$ 个数都是 0.5 ，进位需要加上 $k-1$

```
res = k - 1 + n - (k - 1) / 2;
     = n + (k - 1) / 2;
```

因为 遇 0.5 就进位 故答案可直接写为

```
res = n + k / 2;
```

if $k - 1$ 为奇数

小数出现 0.5 会多进 1 ， $k / 2$ 符合

if $k - 1$ 为偶数

$k / 2$ 向下取整，也符合

故最大值为 $n + k / 2$;

CSDN @_WAWA鱼_

```
1  #include <iostream>
2  #include <algorithm>
3  #include <cstring>
4  #include <vector>
5  using namespace std;
6  void solve()
7  {
8      int n, k; cin >> n >> k;
9      if(n * 2 < k) cout << "0 " << n * 2 << '\n';
10     else cout << n - (k - 1) / 2 << ' ' << n + k / 2 << '\n';
11 }
12
13 signed main()
14 {
15     int T; cin >> T;
16     while(T --) solve();
17 }
18
```

K - 排列与质数 — 构造

构造题没什么可说的，脑筋急转弯，看官方题解吧

对于 $n \leq 10$ ，可以暴力枚举排列求解；

对于 $n > 10$ 的奇数，先将数按照 $1, 3, 5, \dots, n-2, n, n-3, n-5, \dots, 8, 6, 4$ 排列；

对于 $n > 10$ 的偶数，先将数按照 $1, 3, 5, \dots, n-3, n, n-2, n-4, \dots, 8, 6, 4$ 排列；

即先将奇数升序排列，再将偶数降序排列。

CSDN @_WAWA鱼_

可以发现，现在除了 2 和 $n-1$ 以外，所有数均已出现，且满足题目的限制。那么我们只需要将这两个数插进合适的位置即可。容易发现一定有解，因为可以将 2 插在 5 和 7 之间，将 $n-1$ 插在 $n-4$ 和 $n-6$ 之间。

复杂度取决于判断质数的速度， $\mathcal{O}(n\sqrt{n})$ 已经足以通过此题。

CSDN @_WAWA鱼_

```

1  #include <iostream>
2  #include <algorithm>
3  #include <cstring>
4  #include <vector>
5  using namespace std;
6
7  void solve() {
8      int n; cin >> n;
9
10     if (n <= 4) cout << -1 << "\n";
11     else if (n == 5) cout << "4 1 3 5 2\n";
12     else if (n == 6) cout << "1 3 5 2 4 6\n";
13     else if (n == 7) cout << "1 3 5 7 2 4 6\n";
14     else if (n == 8) cout << "1 3 5 7 2 4 6 8\n";
15     else if (n == 9) cout << "1 3 5 7 9 2 4 6 8\n";
16     else if (n == 10) cout << "1 3 10 5 7 9 2 4 6 8\n";
17     else if (n == 11) cout << "1 3 10 5 2 7 9 11 8 6 4\n";
18     else
19     {
20         vector<int> vec;
21         if(n & 1) {
22             for(int i = 1; i <= n; i += 2)
23             {
24                 vec.push_back(i);
25                 if(i == 5) vec.push_back(2);
26                 if(i == n - 6) vec.push_back(n - 1);
27             }
28             for(int i = n - 3; i >= 4; i -= 2)
29                 vec.push_back(i);
30         }
31         else {
32             for(int i = 1; i <= n - 3; i += 2)
33             {
34                 vec.push_back(i);
35                 if(i == 5) vec.push_back(2);
36             }
37             for(int i = n; i >= 4; i -= 2)
38             {
39                 vec.push_back(i);

```

```
40         if(i == n - 4) vec.push_back(n - 1);
41     }
42 }
43 for(auto x : vec) cout << x << " ";
44 cout << '\n';
45 }
46
47 }
48 signed main()
49 {
50     int T; T = 1;
51     while(T --) solve();
52 }
```