

只用暴力，稳拿省一系列。

只允许使用：

判断，循环，数组，函数，语言自带函数，素数算法，gcd，lcm 算法，一维前缀和，一维差分，简单递归。

超出范围的，或需要大量动脑的，我们都不做，只尝试输出样例

握手问题

```
1  #include <iostream>
2  using namespace std;
3  int st[51];
4  int main()
5  {
6      int ans=0;
7      for(int i=1;i<=7;i++) st[i]=1;
8      for(int i=1;i<=50;i++){
9          for(int j=i+1;j<=50;j++){
10             if(!(st[i]&&st[j])){
11                 ans++;
12             }
13         }
14     }
15     printf("%d",ans);
16     return 0;
17 }
```

好数

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  int check(int x){
4      int cnt=1;//维护奇偶数位
5      while(x!=0){
6          int t=x%10;
7          if(cnt%2==1){//奇数
8              if(t%2==0) return 0;
9          }else{
10             if(t%2==1) return 0;
11         }
12         cnt++;
13         x/=10;
14     }
15     return 1;
16 }
17 int main(){
18     int n;
19     scanf("%d",&n);
20     int ans=0;
21     for(int i=1;i<=n;i++){
22         if(check(i)) ans++;
23     }
24     printf("%d",ans);
25 }
```

R 格式

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  int main() {
6      int n;
7      double d;
8      cin >> n >> d;
9
10     // 计算  $d * 2^n$ 
11     double result = d * pow(2, n);
12
13     // 四舍五入到最接近的整数
14     long long r_format = round(result);
15
16     // 输出结果
17     cout << r_format << endl;
18
19     return 0;
20 }
```

宝石组合

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <numeric>
5  using namespace std;
6
7  // 计算最小公倍数
8  long long lcm(long long a, long long b) {
9      return a / gcd(a, b) * b;
10 }
11
12 // 计算精美程度 S
13 double calcs(int a, int b, int c) {
14     long long ab = lcm(a, b);
15     long long ac = lcm(a, c);
16     long long bc = lcm(b, c);
17     long long abc = lcm(ab, c);
18     return 1.0 * a * b * c * abc / (ab * ac * bc);
19 }
20
21 // 找到最大 S 的组合
22 vector<int> findMaxS(int n, vector<int>& h) {
23     double maxS = -1;
24     vector<int> res;
25
26     for (int i = 0; i < n; ++i) {
27         for (int j = i + 1; j < n; ++j) {
28             for (int k = j + 1; k < n; ++k) {
29                 int a = h[i], b = h[j], c = h[k];
```

```

30         double s = calcs(a, b, c);
31
32         if (s > maxS) {
33             maxS = s;
34             res = {a, b, c};
35         } else if (s == maxS) {
36             vector<int> cur = {a, b, c};
37             sort(cur.begin(), cur.end());
38             sort(res.begin(), res.end());
39             if (cur < res) {
40                 res = cur;
41             }
42         }
43     }
44 }
45
46
47 sort(res.begin(), res.end());
48 return res;
49 }
50
51 int main() {
52     int n;
53     cin >> n;
54     vector<int> h(n);
55     for (int i = 0; i < n; ++i) {
56         cin >> h[i];
57     }
58
59     vector<int> ans = findMaxS(n, h);
60     cout << ans[0] << " " << ans[1] << " " << ans[2] << endl;
61
62     return 0;
63 }

```

拔河

```

1  #include <iostream>
2  #include <vector>
3  #include <climits>
4  using namespace std;
5
6  int main() {
7      int n;
8      cin >> n;
9      vector<int> a(n);
10     for (int i = 0; i < n; ++i) {
11         cin >> a[i];
12     }
13
14     // 前缀和数组，方便快速计算区间和
15     vector<int> prefix(n + 1, 0);
16     for (int i = 1; i <= n; ++i) {
17         prefix[i] = prefix[i - 1] + a[i - 1];
18     }
19
20     int min_diff = INT_MAX;

```

```

21
22 // 枚举第一个队伍的区间 [l1, r1]
23 for (int l1 = 0; l1 < n; ++l1) {
24     for (int r1 = l1; r1 < n; ++r1) {
25         int sum1 = prefix[r1 + 1] - prefix[l1];
26
27         // 枚举第二个队伍的区间 [l2, r2], 其中 l2 > r1
28         for (int l2 = r1 + 1; l2 < n; ++l2) {
29             for (int r2 = l2; r2 < n; ++r2) {
30                 int sum2 = prefix[r2 + 1] - prefix[l2];
31
32                 // 计算差距
33                 int diff = abs(sum1 - sum2);
34
35                 // 更新最小差距
36                 if (diff < min_diff) {
37                     min_diff = diff;
38                 }
39             }
40         }
41     }
42 }
43
44 // 输出结果
45 cout << min_diff << endl;
46
47 return 0;
48 }

```