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

```cpp
#include <iostream>
using namespace std;
int st[51];
int main()
{
  int ans=0;
  for(int i=1;i<=7;i++) st[i]=1;
  for(int i=1;i<=50;i++){
    for(int j=i+1;j<=50;j++){
      if(!(st[i]&&st[j])){
        ans++;
      }
    }
  }
  printf("%d",ans);
  return 0;
}
```

好数

```cpp
#include<bits/stdc++.h>
using namespace std;
int check(int x){
  int cnt=1;//维护奇偶数位
  while(x!=0){
    int t=x%10;
    if(cnt%2==1){//奇数
      if(t%2==0) return 0;
    }else{
      if(t%2==1) return 0;
    }
    cnt++;
    x/=10;
  }
  return 1;
}
int main(){
  int n;
  scanf("%d",&n);
  int ans=0;
  for(int i=1;i<=n;i++){
    if(check(i)) ans++;
  }
  printf("%d",ans);
```

```
}
```

R 格式

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    int n;
    double d;
    cin >> n >> d;

    // 计算 d * 2^n
    double result = d * pow(2, n);

    // 四舍五入到最接近的整数
    long long r_format = round(result);

    // 输出结果
    cout << r_format << endl;

    return 0;
}
```

宝石组合

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <numeric>
using namespace std;

// 计算最小公倍数
long long lcm(long long a, long long b) {
    return a / gcd(a, b) * b;
}

// 计算精美程度 S
double calcS(int a, int b, int c) {
    long long ab = lcm(a, b);
    long long ac = lcm(a, c);
    long long bc = lcm(b, c);
    long long abc = lcm(ab, c);
    return 1.0 * a * b * c * abc / (ab * ac * bc);
}

// 找到最大 S 的组合
vector<int> findMaxS(int n, vector<int>& h) {
    double maxS = -1;
    vector<int> res;

    for (int i = 0; i < n; ++i) {
        for (int j = i + 1; j < n; ++j) {
            for (int k = j + 1; k < n; ++k) {
                int a = h[i], b = h[j], c = h[k];
```

```
                double s = calcS(a, b, c);

                if (s > maxS) {
                    maxS = s;
                    res = {a, b, c};
                } else if (s == maxS) {
                    vector<int> cur = {a, b, c};
                    sort(cur.begin(), cur.end());
                    sort(res.begin(), res.end());
                    if (cur < res) {
                        res = cur;
                    }
                }
            }
        }
    }

    sort(res.begin(), res.end());
    return res;
}

int main() {
    int n;
    cin >> n;
    vector<int> h(n);
    for (int i = 0; i < n; ++i) {
        cin >> h[i];
    }

    vector<int> ans = findMaxS(n, h);
    cout << ans[0] << " " << ans[1] << " " << ans[2] << endl;

    return 0;
}
```

拔河

```
#include <iostream>
#include <vector>
#include <climits>
using namespace std;

int main() {
    int n;
    cin >> n;
    vector<int> a(n);
    for (int i = 0; i < n; ++i) {
        cin >> a[i];
    }

    // 前缀和数组，方便快速计算区间和
    vector<int> prefix(n + 1, 0);
    for (int i = 1; i <= n; ++i) {
        prefix[i] = prefix[i - 1] + a[i - 1];
    }

    int min_diff = INT_MAX;
```

```cpp

    // 枚举第一个队伍的区间 [l1, r1]
    for (int l1 = 0; l1 < n; ++l1) {
        for (int r1 = l1; r1 < n; ++r1) {
            int sum1 = prefix[r1 + 1] - prefix[l1];

            // 枚举第二个队伍的区间 [l2, r2]，其中 l2 > r1
            for (int l2 = r1 + 1; l2 < n; ++l2) {
                for (int r2 = l2; r2 < n; ++r2) {
                    int sum2 = prefix[r2 + 1] - prefix[l2];

                    // 计算差距
                    int diff = abs(sum1 - sum2);

                    // 更新最小差距
                    if (diff < min_diff) {
                        min_diff = diff;
                    }
                }
            }
        }
    }

    // 输出结果
    cout << min_diff << endl;

    return 0;
}
```