# Educational Codeforces Round 93 (Rated for Div. 2)

## A. Bad Triangle

**题意：**判断一组排序后的数中是否有三个数可以构成三角形三边，并输出三个数

**思路：**等价于判断最大的数是否可以和最小的两个数构成三角形三边

```
#include <bits/stdc++.h>

using namespace std;
typedef long long LL;
int a[100000];
int main(){
    int t = 0;
    cin>>t;
    while(t--){
        int n;
        scanf("%d", &n);
        for (int i = 1; i <= n; ++i) {
            scanf("%d", a+i);
        }

        if(a[1] + a[2] <= a[n])printf("1 2 %d\n", n);
        else printf("-1\n");
    }
}
```

## B. Substring Removal Game

```
#include <bits/stdc++.h>

using namespace std;
typedef long long LL;

int num[1005], tot;

int main() {
    int t;
    cin >> t;
    while (t--) {
        string s;
        cin >> s;
```

```
        tot = 0;
        int tt = 0;
        int n = s.length();
        for (int i = 0; i < n; i++) {
            if (s[i] == '1') {
                tt++;
                continue;
            }
            if (tt != 0) {
                num[tot] = tt;
                tot++;
                tt = 0;
            }
        }
        if (tt != 0) {
            num[tot] = tt;
            tot++;
        }
        sort(num, num + tot);
        int ans = 0;
        for (int i = tot - 1; i >= 0; i -= 2) {
            ans += num[i];
        }
        printf("%d\n", ans);
    }
    return 0;
}
```

# C. Bad Triangle

**题意：**求子串内字符值和与子串下标差相等的子串个数

**思路：**求前缀和后，减去当前下标，符合要求的子串的首位下表对应的值相等

```
#include <bits/stdc++.h>

using namespace std;
typedef long long LL;
int a[100005];
int sum[100005];
int cnt[2000005];

int main() {
    int t = 0;
    cin >> t;
    while (t--) {
        int n;
        cin >> n;
```

```
        string s;
        cin >> s;
        for (int i = 0; i < n; ++i) {
            a[i] = s[i] - '0';
        }
        sum[0] = a[0];
        for (int j = 1; j < n; ++j) {
            sum[j] = sum[j - 1] + a[j];
        }
        for (int i = 0; i < n; ++i) {
            sum[i] -= i + 1;
        }
        memset(cnt, 0, sizeof(int) * n * 10 * 2 + 2);
        cnt[0 + n * 10] = 1;
        LL ans = 0;
        for (int i = 0; i < n; ++i) {
            if(cnt[sum[i] + n * 10]){
                ans += cnt[sum[i] + n * 10];
            }
            cnt[sum[i] + n * 10]++;
        }
        cout<<ans<<endl;
    }
}
```

# D. Colored Rectangles

**思路：**dp□dp[i][j][k]表示r用前i个、g用前j个，b用前k个组成的长方形面积最大值，考虑r□g长方形增设的情况，可由i-1 j-1 k的结果加上剩下r□g边最大值相乘得到。

```
#include <bits/stdc++.h>

using namespace std;
typedef long long ll;

const int maxn = 220;
int r[maxn], g[maxn], b[maxn];

ll dp[maxn][maxn][maxn];

int main() {
    int R, G, B;
    scanf("%d%d%d", &R, &G, &B);
    for (int i = 0; i < R; i++) scanf("%d", &r[i]);
    for (int i = 0; i < G; i++) scanf("%d", &g[i]);
    for (int i = 0; i < B; i++) scanf("%d", &b[i]);

    sort(r, r + R, greater<int>());
```

```
    sort(g, g + G, greater<int>());
    sort(b, b + B, greater<int>());

    ll ans = -1;

    for (int i = 0; i <= R; i++)
        for (int j = 0; j <= G; j++)
            for (int k = 0; k <= B; k++) {
                dp[i + 1][j + 1][k] = max(dp[i + 1][j + 1][k], dp[i][j][k]
+ 1ll * r[i] * g[j]);
                dp[i + 1][j][k + 1] = max(dp[i + 1][j][k + 1], dp[i][j][k]
+ 1ll * r[i] * b[k]);
                dp[i][j + 1][k + 1] = max(dp[i][j + 1][k + 1], dp[i][j][k]
+ 1ll * g[j] * b[k]);
                ans = max({ans, dp[i + 1][j + 1][k], dp[i + 1][j][k + 1],
dp[i][j + 1][k + 1]});
            }

    printf("%lld\n", ans);
    return 0;
}
```