Non-Divisible Subset

by zxqfd555

Editorial by zxqfd555

First, let's count the number of integers having every remainder of division by k (i.e., 0 through $k-1$). Let's denote the number of integers from the set which give the remainder t modulo k as $A[t]$.

Then, consider some specific remainder $t > 0$. If we take at least one integer with the remainder t and at least one with the remainder $k-t$, then the sum of these two integers will be evenly divisible by k. Therefore, for any fixed t we'll have to decide what to take to the answer set: $A[t]$ integers with the remainder t, or $A[k-t]$ integers with the remainder $k-t$. We choose whichever value is greater.

The above works except for two cases:

- For $t = 0$ there's no different *pair* remainder which would have a sum evenly divisible by k, but we also can't take 2 or more numbers with the remainder equal to 0 because their sum would be evenly divisible by k. So we should only add $\min(1, A[0])$ to our answer.
- Also, if k is even and $t = \frac{k}{2}$, then taking two integers with the remainder t will make the sum divisible by k. So for even k and $t = \frac{k}{2}$ we should take $\min(1, A[\frac{k}{2}])$.

Set by zxqfd555

Problem Setter's code :

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

using namespace std;

const int MAXM = 100;

int n, m, st[MAXM], sp, sz, a[MAXM], ret, tn, ai;
bool used[MAXM];
int forbidden[MAXM];
set<int> S;

int main() {
    cin >> n >> m;
    for(int i = 1; i <= n; i++) {
        cin >> ai;
        ++a[ai % m];
        S.insert(ai);
    }
    if (m % 2 == 0)
        a[m / 2] = min(a[m / 2], 1);
    ret = 0;
    for(int i = 1; i <= m / 2; i++)
        ret += max(a[i], a[m - i]);
    ret += min(a[0], 1);
    cout << ret << endl;
    return 0;
}
```

Statistics

Difficulty: Medium
Time Complexity: $\mathcal{O}(n+k)$
Required Knowledge: Arrays, modulo
Publish Date: May 22 2016

Originally featured in Week of Code - 20

Tested by shef_2318

Problem Tester's code :

```
#include <bits/stdc++.h>
using namespace std;

const int MAXK = 100, MAXT = 100, MAXN = 100000, MAXA = 1E9;
int cnt[110], n, k;
void solve() {
    memset(cnt, 0, sizeof(cnt) );
    scanf("%d%d", &n, &k);
    for (int i = 0; i < n; i++) {
        int x;
        scanf("%d", &x);
        x %= k;
        cnt[x]++;
    }
    int ans = 0;
    ans += min(1, cnt[0]);
    for (int i = 1; i < k/2 + k%2; i++) {
        ans += max(cnt[i], cnt[k - i]);
    }
    if (k % 2 == 0) {
        ans += min(1, cnt[k/2]);
    }
    cout<<ans<<endl;
}

int main() {
    int cases = 1;
    for (int i = 0; i < cases; i++) {
        solve();
    }
    return 0;
}
```