1828A - Divisible Array

Idea: **thenymphsofdelphi**

Preparation: **Mike4235**

Hint 1

Remember the sum of the first n positive integers?

Hint 2

Every positive integer is divisible by 1.

Solution

Consider the array $a = [1, 2, \ldots, n]$ that satisfies the second condition. It has the sum of $1 + 2 + \cdots + n = \frac{n(n+1)}{2}$.

One solution is to notice that if we double every element $(a = [2, 4, 6, \ldots, 2n])$, the sum becomes $\frac{n(n+1)}{2} \times 2 = n(n+1)$, which is divisible by n.

Another solution is to increase the value of a_1 until the sum becomes divisible by n. This works because every integer is divisible by 1, and we only need to increase a_1 by at most n.

Time complexity: $O(n)$

Implementation 1

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

using namespace std;

int main() {

    ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);

    int t;
    cin >> t;
    while (t--) {
        int n;
        cin >> n;
        for (int i = 1; i <= n; i++) cout << i * 2 << " ";
        cout << "\n";
    }
}
```

Implementation 2

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
#define fi first
#define se second
```

```
const int N=2e6+1;
const ll mod=998244353;
ll n,m;
ll a[N],b[N];
void solve(){
        cin >> n;
        ll s=0;
        for(int i=n; i>=2 ;i--){
                a[i]=i;
                s=(s+i)%n;
        }
        a[1]=n-s;
        for(int i=1; i<=n ;i++) cout << a[i] << ' ';
        cout << '\n';
}
int main(){
        ios::sync_with_stdio(false);cin.tie(0);
        int t;cin >> t;
        while(t--){
                solve();
        }
}
```