Problem: Non-Divisible Subsets

Given a set, , of distinct integers, print the size of a maximal subset, , of where the sum of any numbers in is *not* evenly divisible by .

Input Format

The first line contains space-separated integers, and , respectively.
The second line contains space-separated integers (we'll refer to the value as) describing the unique values of the set.

Constraints

-
-
-
- All of the given numbers are distinct.

Output Format

Print the size of the largest possible subset ().

Sample Input

```
4 3
1 7 2 4
```

Sample Output

```
3
```

Explanation

The largest possible subset of integers is , because no two integers will have a sum that is evenly divisible by :

- , and is not evenly divisible by .
- , and is not evenly divisible by .
- , and is not evenly divisible by .

The number cannot be included in our subset because it will produce an integer that is evenly divisible by when summed with any of the other integers in our set:

- , and .
- , and .
- , and .

Thus, we print the length of on a new line, which is .

Solution

```
int main()
   {
       /*Feeding the data*/
       long cases, k;
       cin>>cases >>k;
       long values[cases];
       int possible[k]={0};

       for(long i=0; i<cases; i++)
          {long value;
           cin>>value;
           possible[value%k]+=1;
          }


       /*Processing the possible array*/
       long sum=min(1, possible[0]);
       long middle=k/2;
       if(k%2==0)
          {  //cout<<"Inside even case          // 123 4 567
              for(long i=1; i<middle; i++)
                 { sum+=max(possible[i],possible[k-i]);
                 }
              sum+=min(1, possible[middle]);
              cout<<sum;
          }
       else
          {  //cout<<"Inside odd case";   //1234
              for(long i=1; i<=middle; i++)
              {
                 sum+=max(possible[i],possible[k-i]);
              }
          cout<<sum;
          }
       return 0;
 }                                                - Anshul Aggarwal
```