COCI 2009/2010 **Task PROGRAM**

5th round, 6. March 2010. 5 seconds / 32 MB / 120 points

Mirko is trying to debug a piece of his code. First he creates an array of N integers and fills it with zeros. Then he repeatedly calls the following procedure (he is such a good coder he coded it in both C++ and Pascal):

```
void something( int jump ) {
   int i = 0;
   while( i < N ) {
      seq[i] = seq[i] + 1;
      i = i + jump;
   }
}
```

```
procedure something( jump: longint );
var i : longint;
begin
  i := 0;
  while i < N do
  begin
    seq[i] := seq[i] + 1;
    i := i + jump;
  end;
end;
```

As you can see, this procedure increases by one all elements in the array whose indices are divisible by `jump`.

Mirko calls the procedure exactly **K** times, using the sequence $\mathbf{X_1}\ \mathbf{X_2}\ \mathbf{X_3}\ldots\ \mathbf{X_k}$ as arguments.

After this, Mirko has a list of **Q** special parts of the array he needs to check to verify that his code is working as it should be. Each of this parts is defined by two numbers, **L** and **R** (**L** ≤ **R**) the left and right bound of the special part. To check the code, Mirko must compute the sum of all elements of `seq` between and including **L** and **R**. In other words `seq`[**L**] + `seq`[**L**+1] + `seq`[**L**+2] + ... + `seq`[**R**]. Since he needs to know the answer in advance in order to check it, he asked you to help him.

INPUT

The first line of input contains two integers, **N** ($1 \le \mathbf{N} \le 10^6$), size of the array, and **K** ($1 \le \mathbf{K} \le 10^6$), number of calls to `something` Mirko makes.

The second line contains **K** integers: $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3$ ... $\mathbf{X}_k$, arguments passed to the procedure. ($1 \leq \mathbf{X}_i < \mathbf{N}$).
Next line contains one integer **Q** ($1 \leq \mathbf{Q} \leq 10^6$), number of special parts of the array Mirko needs to check.
Next **Q** lines contain two integers each $\mathbf{L}_i$ i $\mathbf{R}_i$($0 \leq \mathbf{L}_i \leq \mathbf{R}_i < \mathbf{N}$), bounds of each special part.

OUTPUT

The output should contain exactly **Q** lines. The i^{th} line should contain the sum of elements seq[$\mathbf{L}_i$] + seq[$\mathbf{L}_i$+1] + seq[$\mathbf{L}_i$+2] + ... + seq[$\mathbf{R}_i$].

SAMPLE TEST CASES

Input:	**Input:**	**Input:**
10 4 1 1 2 1 3 0 9 2 6 7 7	11 3 3 7 10 3 0 10 2 6 7 7	1000000 6 12 3 21 436 2 19 2 12 16124 692 29021
Output:	**Output:**	**Output:**
35 18 3	8 2 1	16422 28874

Sample 1. description: The procedure is called with arguments 1, 1, 2, 1. After that the array contains values {4, 3, 4, 3, 4, 3, 4, 3, 4, 3}. Sum of indices 2 to 6 (inclusive) is 4+3+4+3+4 = 18.

Sample 2. description: The array is {3, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1}.