Range Minimum Query

- Given a sequence of n integers a_0,. . ., a_{n-1}. We denote rmq(i, j) the minimum element of the sequence a_i, a_{i+1}, . . ., a_j. Given m pairs (i_1, j_1),. . ., (i_m, j_m), compute the sum Q = rmq(i_1, j_1) + . . . + rmq(i_m, j_m)
- **Input**
  - Line 1: contains an integer n ($1 <= n <= 10^6$)
  - Line 2: contains a_0, . . . , a_{n-1} ($1 <= a_i <= 10^6$)
  - Line 3: contains m ($1 <= m <= 10^6$)
  - Line k+3 (k = 1, . . ., m): contains i_k, j_k ($0 <= i_k < j_k < n$)
- **Output**
  - Write the value Q

Hint

- Denote $M[j, i]$ the index of the smallest element of $a[i]$, $a[i+2]$,…, $a[i+2^j-1]$ (the sequence from index i and has the length 2^j).

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	**4**	**6**	**1**	**6**	**8**	**7**	**3**	**3**	**5**	**8**	**9**	**1**	**2**	**6**	**4**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	0	1	3	3	4	6	7	8	8	9	10	12	12	13	15	-
2	3	3	3	3	7	8	8	8	8	12	12	12	12	-	-	-
3	3	3	3	3	8	12	12	12	12	-	-	-	-	-	-	-
4	12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

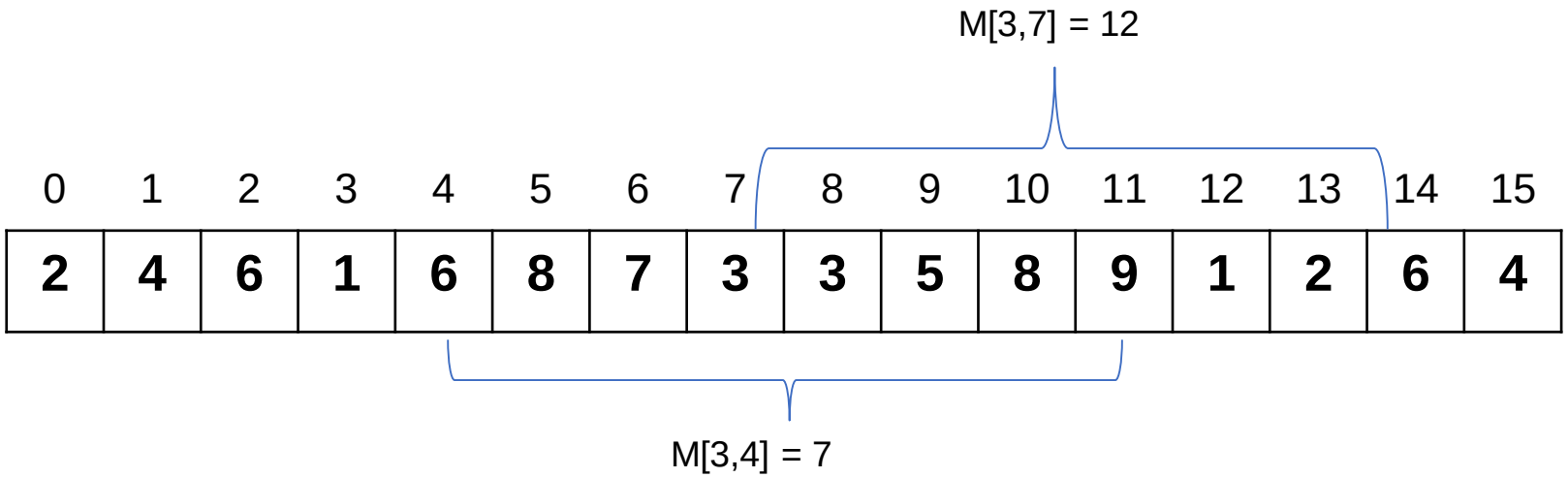
Range Minimum Query

- Example

stdin	stdout
16 2 4 6 1 6 8 7 3 3 5 8 9 1 2 6 4 4 1 5 0 9 1 15 6 10	6

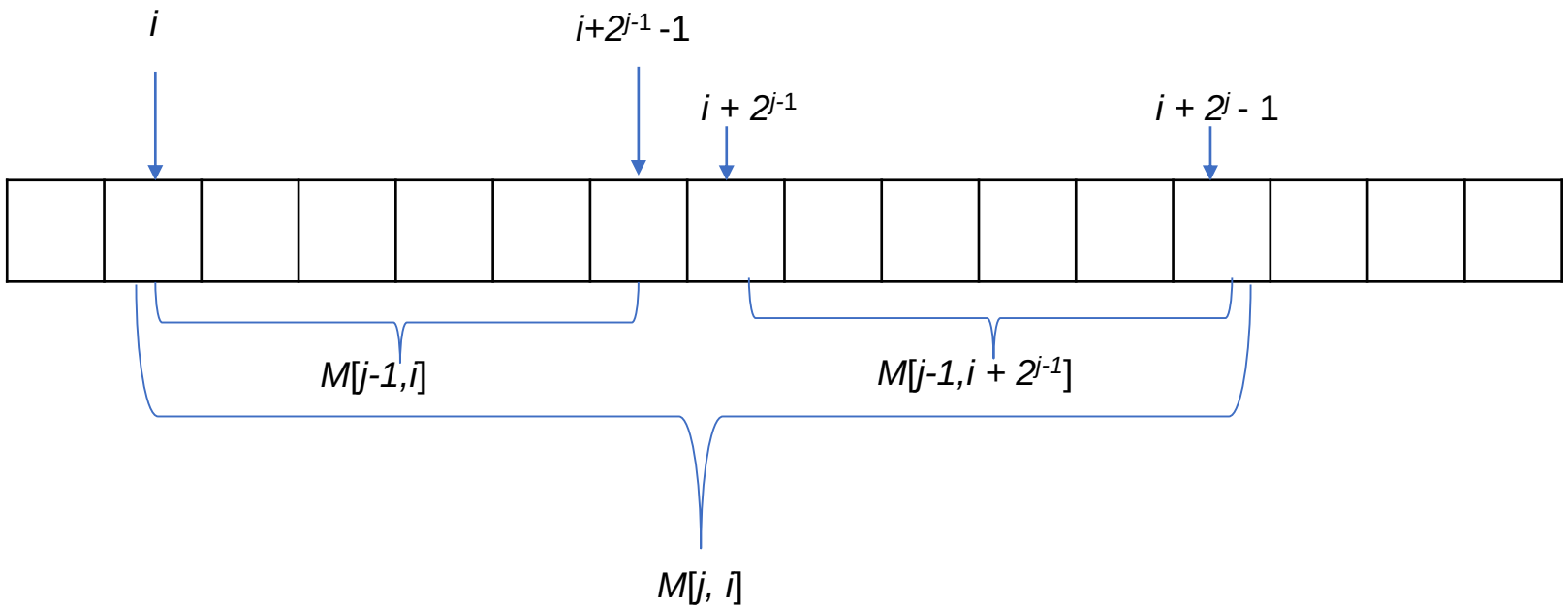
Hint

- Query RMQ(i,j): the index of the smallest element of the sequence $a[i]$, $a[i+1]$, . . ., $a[j]$
- $k = [\log(j-i+1)]$
- RMQ(i,j) = $M[k,i]$ if $a[M[k,i]] \leq a[M[k, j-2^k+1]]$
  $M[k, j-2^k+1]]$, otherwise
- RMQ(4,14) = ?
  - $k = [\log(14-4+1)]=3$
  - $a[7] > a[12]$ → RMQ(4,14) = 12



Hint

- $M[0,i] = i,\ i = 0,\ldots,\ N-1$
- Recurrence relation:
- $M[j,i] = \begin{cases} M[j-1,i] \text{ if } a[M[j-1,i]] < a[M[j-1,i+2^{j-1}] \\ M[j-1,i+2^{j-1}], \text{ otherwise} \end{cases}$



Implementation

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

using namespace std;

int n;

int M[30][1000000];

int A[1000000];

void preprocessing(){

    for(int j = 0; (1 << j) <= n; j++){

        for(int i = 0; i < n; i++) M[j][i] = -1;

    }

    for(int i = 0; i < n; i++) M[0][i] = i;

    for(int j = 1; (1 << j) <= n; j++){

        for(int i = 0; i + (1 << j) - 1 < n; i++){

            if(A[M[j-1][i]] < A[M[j-1][i+(1 << (j-1))]])  M[j][i] = M[j-1][i];  else  M[j][i] = M[j-1][i + (1 << (j-1))];

        }

    }

}
```

Implementation

```
int rmq(int i, int j){
    int k = log2(j-i+1);
    int p2k = (1 << k);//pow(2,k);
    if(A[M[k][i]] <= A[M[k][j-p2k+1]]){
        return M[k][i];
    }else{
        return M[k][j-p2k+1];
    }
}
```

Implementation

```
int main(){
   scanf("%d",&n);
   for(int i = 0; i < n; i++)   scanf("%d",&A[i]);


   preprocessing();


   int ans = 0; int m;
    scanf("%d",&m);
    for(int i = 0; i < m; i++){
        int I,J;  scanf("%d%d",&I,&J);
        ans += A[rmq(I,J)];
    }
    cout << ans;
    return 0;
}
```