XOR key

Xorq has invented an encryption algorithm which uses bitwise XOR operations extensively. This encryption algorithm uses a sequence of non-negative integers $x_1, x_2, \cdots x_n$ as key. To implement this algorithm efficiently, *Xorq* needs to find maximum value of $(a \oplus x_j)$ for given integers a, p and q, such that, $p \leqslant j \leqslant q$. Help *Xorq* implement this function.

Input Format

First line of input contains the number of test cases, T (1<=T<=6). T test cases follow.
First line of each test case contains two space separated integers N and Q (1<= N<=100,000; 1<=Q<= 50,000). Next line contains N space separated integers $x_1, x_2, \cdots x_n$ (0<=x_i< 2^{15}). Each of next Q lines describes a query which consists of three integers a_i, p_i and q_i (0<=a_i< 2^{15}, 1<=p_i<=q_i<= N).

Output Format

For each query, print in a new line the maximum value for $(a_i \oplus x_j)$, such that, $p_i \leqslant j \leqslant q_i$.

Sample Input

```
1
15 8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
10 6 10
1023 7 7
33 5 8
182 5 10
181 1 13
5 10 15
99 8 9
33 10 14
```

Sample Output

```
13
1016
41
191
191
15
107
47
```

Explanation

- First Query (10 6 10): $x_6 \oplus 10 = 12, x_7 \oplus 10 = 13, x_8 \oplus 10 = 2, x_9 \oplus 10 = 3, x_10 \oplus 10 = 0$, therefore answer for this query is 13.
- Second Query (1023 7 7): $x_7 \oplus 1023 = 1016$, therefore answer for this query is 1016.
- Third Query (33 5 8): $x_5 \oplus 33 = 36, x_6 \oplus 33 = 39, x_7 \oplus 33 = 38, x_8 \oplus 33 = 41$, therefore answer for this query is 41.
- Fourth Query (182 5 10): $x_5 \oplus 182 = 179, x_6 \oplus 182 = 176, x_7 \oplus 182 = 177, x_8 \oplus 182 = 190, x_9 \oplus 182 = 191, x_{10} \oplus 182 = 188$, therefore answer for this query is 191.