一、简答

1. 使用位运算：

```
(a & (1 << i)) ? a - (1 << i) : a + (1 << i)
```

2. 所有元素入栈或入队一次，所以它们是线性复杂度的。

二、作业H5

E.旅途不止

题目大意

有一列长度为 n 的数，初始值都是 1 。有 m 次操作，每次对属于区间 [l, r] 的数都乘上一个数 c^b ，最后输出这 n 个数的最大公约数。

解法

1. c 的数据范围较小，可以为转为成维护100以内素数的幂。对于每个底数，通过差分数组优化区间加。这样最后答案为

$$\prod_{c\text{为}100\text{以内素数}} \left(c^{\min_{j\in[1,100]}(b_j)}\right)$$

2. 使用快速幂进行幂运算。

时间复杂度

$O(25(m+n))$

代码

```cpp
#include<bits/stdc++.h>
using namespace std;
using ll = long long;
const int Mod = 1e9 + 7;

ll d[25][100005];
int pr[] = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79,
/*
a0 a1 a2 a3 a4
d1 = a1
d4 = a4 - a3
*/

ll qpow(ll a, ll b) {
    ll res = 1;
    while (b) {
        if (b % 2) res = res * a % Mod;
        a = a * a % Mod;
        b /= 2;
    }
    return res;
}

void solve() {
    int n, m;
    cin >> n >> m;

    int l, r, c, b;
    for (int i = 1; i <= m; i++) {
        cin >> l >> r >> c >> b;
        for (int j = 0; j < 25; j++) {
            if (!(c % pr[j])) {
                d[j][r + 1] -= b;
                d[j][l] += b;
                c /= pr[j];
                j--;
            }
        }

    }

    ll p = 0, Min, ans = 1;
    for (int i = 0; i < 25; i++) {
        Min = LONG_LONG_MAX;
        p = 0;
        for (int j = 1; j <= n; j++) {
            p += d[i][j];
            // cout << p << " ";
            Min = min(Min, p);
        }
```

```
        // cout << '\n';
        ans = ans * qpow(pr[i], Min) % Mod;
    }

    cout << ans << '\n';
}

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    solve();
    return 0;
}
```