Sparse table for (min / max / gcd) query in 2D-Array:

```cpp
int a[1010][1010], st[1010][1010][15][15] , lg2[1010];

//build the table in O(n*m*log(n)*log(m))
//change every min to max in case of max query
void build(int n, int m){
    //fill the log array
    for (int i = 2; i < 1000; i++)
        lg2[i] = lg2[i >> 1] + 1;

    //fill base case in st
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            st[i][j][0][0] = a[i][j];

    //build the st
    for (int l1 = 0; l1 < 15; l1++) {
        for (int l2 = 0; l2 < 15; l2++) {
            if (l1 + l2 == 0) continue;
            for (int i = 0; i + (1 << l1) <= n; i++) {
                for (int j = 0; j + (1 << l2) <= m; j++) {
                    if (!l1){
                        st[i][j][l1][l2] = min( st[i][j][l1][l2 - 1] ,
                                        st[i][j + (1 << (l2 - 1))][l1][l2 - 1] );
                    }
                    else{
                        st[i][j][l1][l2] = min( st[i][j][l1 - 1][l2] ,
                                        st[i + (1 << (l1 - 1))][j][l1 - 1][l2] );
                    }
                }
            }
        }
    }
}

//get query in O(1)
int get(int x1, int y1, int x2, int y2) {
    x2++; y2++;
    int l1 = lg2[x2 - x1], l2 = lg2[y2 - y1];
    return min(
            min(st[x1][y1][l1][l2], st[x2 - (1 << l1)][y1][l1][l2]),
            min(st[x1][y2 - (1 << l2)][l1][l2] ,
                st[x2 - (1 << l1)][y2 - (1 << l2)][l1][l2] )
        );
}
```