Estrutura de dados

Fenwick Tree

Resolve queries do tipo RSQ de 1 a n (1-indexed) em O(log n).

A arvore é contruida inicialmente zerada e para relizar a inserção dos itens deve ser feita a soma ao indice i do valor x

```
template <typename T>
class BITree {
private:
    vector<T> ts;
    size_t N;

    int LSB(int n) { return n&(-n); }

    T RSQ(int i){
        T sum = 0;

        while(i>=1){
            sum+=ts[i];
            i-=LSB(i);
        }
        return sum;
    }
public:
    BITree(size_t n) : ts(n+1, 0), N(n) {};

    T RSQ(int i, int j){
        return RSQ(j) - RSQ(i-1);
    }

    void add(size_t i, const T& x){
        if(i==0) return;
        while(i<=N){
            ts[i]+=x;
            i+=LSB(i);
        }
    }
};
```

Fenwick Tree bidimensional

```
template<typename T>
class BITree2D{
private:
    size_t rows;
    size_t columns;
    vector<vector<T>> ft;

    int LSB(T n) { return n&(-n); }

    T RSQ(int y, int x){
        T sum = 0;

        for(int i=y; i>0; i-=LSB(i))
            for(int j=x; j>0; j-=LSB(j))
                sum+=ft[i][j];

        return sum;
    }
public:
    BITree2D(size_t y, size_t x) : ft(y+1, vector<T>(x+1, 0)), rows(y),
columns(x) {}

    // Y e X sao as coordenadas do ponto superior
    // x e y sao as coordenadas do ponto inferior
    // RSQ vai retornar a soma do quadrado formado pelos pontos
    T RSQ(int y, int x, int Y, int X){
        return RSQ(Y, X) - RSQ(Y, x-1) - RSQ(y-1, X) + RSQ(y-1, x-1);
    }

    void add(int y, int x, T v){
        for(int i=y; i<=rows; i+=LSB(i))
            for(int j=x; j<=columns; j+=LSB(j))
                ft[i][j] += v;
    }

};
```