A5. Поиск значения в отсортированной матрице

1.Разработайте линейный алгоритм поиска значения key в заданной матрице. Представьте описание алгоритма любым удобным способом – псевдокод, блок-схема, код на C++ и пр.

идея моего решения такова:

идем по диагонали, пока не встретим элемент который больше чем k, если встретим k то вернем его

когда встретили диагональный элемент больше нашего, бежим вниз и влево пока не найдем наше значение

```
#include <vector>

std::pair<int,int> index_by_key(std::vector<std::vector<int>> matrix, int
n, int k){
    for(size_t i = 0; i < n; ++i){
        if(matrix[i][i] == k){
            return std::make_pair(i,i);
        }
        if(matrix[i][i] > k){
            for(size_t j = i - 1; j >= 0; --j){
                if(matrix[i][j] == k){
                    return std::make_pair(i,j);
                }
                if(matrix[i][j] < k){
                    break;
                }
            }

            for(size_t j = i - 1; j >= 0; --j){
                if(matrix[j][i] == k){
                    return std::make_pair(j,i);
                }
                if(matrix[j][i] < k){
                    break;
                }
            }
            break;
        }
    }
    return std::make_pair(-1,-1);
}
```

код можно посмотреть в файле

2. Выполните анализ временной сложности разработанного алгоритма, составив точное выражение функции временной сложности $T(N)$. Докажите, что $T(N) = O(N)$ в соответствии с определением асимптотической верхней границы.

в худшем случае, мы пробежим всю диагональ, верхнюю строку и правый столбец, и вернем `(-1,-1)`

итого $T(N) = 5N + 8(N-1)+8(N-1)+3=21N-13$

докажем что $T(N) = O(N)$

$21N-13<CN$

для $C=21$ работает для всех N

ч.т.д.