A5. Поиск значения в отсортированной матрице

Демченко Георгий Павлович , БПИ-235

1. FindKeyInSortedMatrix.cpp

```cpp
#include <vector>
#include <cstdint>
#include <utility>

std::pair<int32_t , int32_t > FindKeyInSortedMatrix(const
std::vector<std::vector<int32_t>>& matrix, int32_t key) {
    int32_t rowIdx = 0;
    int32_t columnIdx = 0;
    size_t n = matrix.size();

    while (rowIdx < n && columnIdx < n) {
        if (matrix[rowIdx][columnIdx] == key) {
            return std::make_pair(rowIdx, columnIdx);
        }
        if (matrix[rowIdx][columnIdx] < key) {
            columnIdx++;
        } else if (matrix[rowIdx][columnIdx] > key) {
            rowIdx++;
        }
    }

    throw std::runtime_error("No such key in matrix");
}
```

2. Анализ временной сложности FindKeyInSortedMatrix.cpp

Первые 3 строки - присваивание $\Rightarrow (c_1 + c_2 + c_3)$

- На каждой итерации цикла **while** может происходить:
  - 3 элементарых операции (1ое сравнение -> удволетворение -> создание пары и возврат)
  - 3 элементарных операции (1ое и 2ое сравнение -> удволетворение -> инкремент)
  - 4 элементарных операции (1, 2, 3 сравнение -> удволетворение -> инкремент)

Так как на произвольной итерации невозможно отследить ветвление, то будем считать, что на каждой итерации происходило 3 элементарных операции $\Rightarrow (c_4 + c_5 + c_6)$

Если число key присутствует в матрице:

- $\text{keyRIdx} \in [0; n - 1]$ - индекс строки , где расположен key
- $\text{keyCIdx} \in [0; n - 1]$ - индес столбца, где расположен key

То произойдет в точности $(\text{keyRIdx} + \text{keyCId} + 1) \leq 2n - 1$ итераций

И $(\text{keyRIdx} + \text{keyCId}) \leq 2n$ сравнений для входа в цикл

$\Rightarrow T(n) = (\text{keyRIdx} + \text{keyCId} + 1) \cdot (c_4 + c_5 + c_6) + (\text{keyRIdx} + \text{keyCId}) \cdot c_7 + (c_1 + c_2 + c_3)$

$= (\text{keyRIdx} + \text{keyCId}) \cdot (c_4 + c_5 + c_6 + c_7) + (c_1 + c_2 + c_3 + c_4 + c_5 + c_6)$

Если число key отстутствует в матрице:

- Если $\text{key} > \max(\text{matrix})$, то произойдет n итераций и $n + 1$ сравнение на вход

$\Rightarrow T(n) = n(c_4 + c_5 + c_6) + (n + 1) \cdot c_7 + (c_1 + c_2 + c_3) = n(c_4 + c_5 + c_6 + c_7) + (c_1 + c_2 + c_3 + c_7)$

- Если $\text{key} < \min(\text{matrix})$, то произойдет n итераций и $n + 1$ сравнение на вход

$\Rightarrow T(n) = n(c_4 + c_5 + c_6) + (n + 1) \cdot c_7 + (c_1 + c_2 + c_3) = n(c_4 + c_5 + c_6 + c_7) + (c_1 + c_2 + c_3 + c_7)$

- Иначе в худшем случае (найти общий случай не является возможным) произойдет $2n - 1$ итераций и $2n$ сравнений на вход

$\Rightarrow T(n) = (2n - 1)(c_4 + c_5 + c_6) + 2n \cdot c_7 + (c_1 + c_2 + c_3) = 2n(c_4 + c_5 + c_6 + c_7) + (c_1 + c_2 + c_3 - c_4 - c_5 - c_6)$

· $T(n) = O(n)$

Будем рассматировать случай когда key присутсвует в матрице (остальные случаи аналогично линейны как показано выше)

- $\text{keyRIdx} \in [0; n - 1]$ - индекс строки , где расположен key
- $\text{keyCIdx} \in [0; n - 1]$ - индес столбца, где расположен key
- $T(n) = (\text{keyRIdx} + \text{keyCId}) \cdot (c_4 + c_5 + c_6 + c_7) + (c_1 + c_2 + c_3 + c_4 + c_5 + c_6)$

Произведем замены для красоты:

- $(c_4 + c_5 + c_6 + c_7) = c_8$
- $(c_1 + c_2 + c_3 + c_4 + c_5 + c_6) = c_9$

$\Rightarrow T(n) = (\text{keyRIdx} + \text{keyCId}) \cdot c_8 + c_9 \leq (2n - 2) \cdot c_8 + c_9 = 2n \cdot c_8 + (c_9 - 2c_8)$

$\Rightarrow \exists\, c_{100} \in \mathbb{R}^+ : \forall \mathbb{N} > N_0 : (\text{keyRIdx} + \text{keyCId}) \cdot c_8 + c_9 \leq 2n \cdot c_8 + (c_9 - 2c_8) \leq c_{100} \cdot n$

- Пусть $N_0 = 1$

$c_{100} \geq 2c_8 + \frac{c_9}{n} - \frac{2c_8}{n}$

$\Rightarrow c_{100} \geq (2c_8 + \frac{c_9}{2}) \in \mathbb{R}^+$

$\Rightarrow T(n) = O(n)$