Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по по домашнему заданию

по курсу «Анализ Алгоритмов»

на тему: «Графовые модели алгоритмов»

Студент ИУ7-53Б (Группа) _____ (Подпись, дата) Лысцев Н. Д. (И. О. Фамилия)

Преподаватель _____ (Подпись, дата) Волкова Л. Л. (И. О. Фамилия)

2024 г.

СОДЕРЖАНИЕ

1 Графовые модели алгоритмов

Программа представлена в виде графа — набора вершин и множества соединяющих их дуг. Вершины — операторы, срабатывания операторов. Дуги — отношения.

Существует два типа отношений:

- информационное отношение — отношение по передаче данных;
- операционное отношение — отношение по передаче управления.

Выделяют четыре графовых модели:

- `граф управления` — модель, в которой вершинами являются операторы, а дугами — операционные отношения;
- `информационный граф` — модель, в которой вершинами являются операторы, а дугами — информационные отношения;
- `операционная история` — модель, в которой вершинами являются срабатывания операторов, а дугами — операционные отношения;
- `информационная история` — модель, в которой вершинами являются срабатывания операторов, а дугами — информационные отношения.

2 Выполнение задания

2.1 Средства реализации

В качестве языка программирования был выбран $C++$ [1].

2.2 Код программы

В листинге 2.1 и 2.2 представлена реализация алгоритма составления файла словаря с количеством употреблений каждой N-граммы букв из одного слова в тексте на русском языке.

Листинг 2.1 – Реализация алгоритма составления файла словаря с количеством употреблений каждой N-граммы букв из одного слова в тексте на русском языке (начало)

```
void processText(std::vector<std::wstring> &vecStrText, const int
   ngram, std::map<std::wstring, int> &ngramCounts)
{
    for (int i = 0; i < (int)vecStrText.size(); ++i) // 1
    {
        size_t startPos = 0; // 2
        size_t endPos = 0;   // 3

        while (endPos != std::wstring::npos) // 4
        {
            endPos = vecStrText[i].find(L' ', startPos); // 5

            std::wstring word = vecStrText[i].substr(startPos,
               endPos - startPos); // 6

            if (static_cast<int>(word.size()) < ngram) // 7
            {
                startPos = endPos + 1; // 8
                continue;              // 9
            }

            std::vector<std::wstring> ngrams; // 10
```

Листинг 2.2 – Реализация алгоритма составления файла словаря с количеством употреблений каждой N-граммы букв из одного слова в тексте на русском языке (конец)

```
            for (size_t i = 0; i <= word.length() - ngram; ++i) //
               11
                ngrams.push_back(word.substr(i, ngram));        //
                   12

            for (const auto &ngram : ngrams) // 13
                ngramCounts[ngram]++;        // 14

            startPos = endPos + 1; // 15
        }
    }
}
```

2.3 Графовые модели программы

На рисунке 2.1 представлен граф управления.

На рисунке 2.2 представлен информационный граф.

На рисунках 2.3 и 2.4 представлена операционная история.

На рисунках 2.5 и 2.6 представлена информационная история.

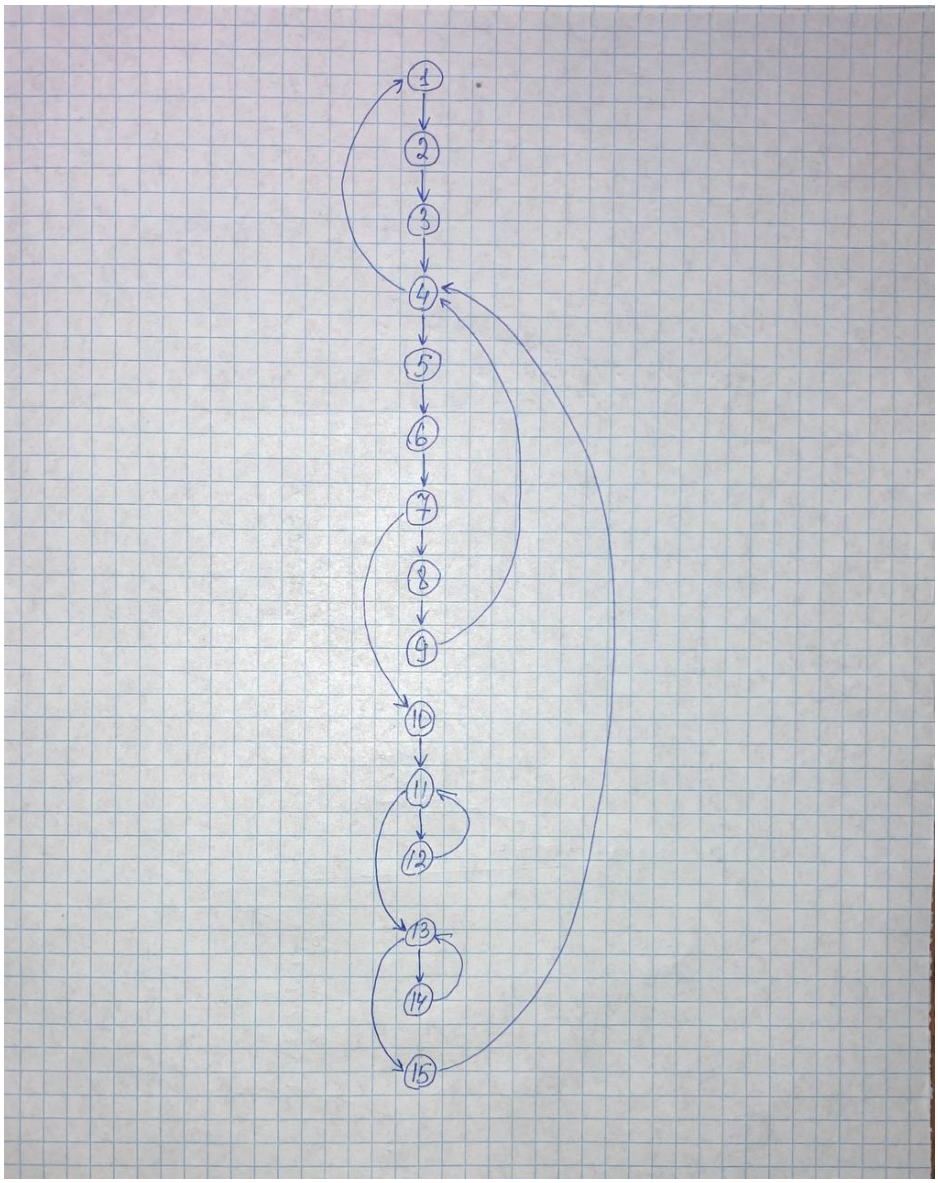

Рисунок 2.1 – Граф управления

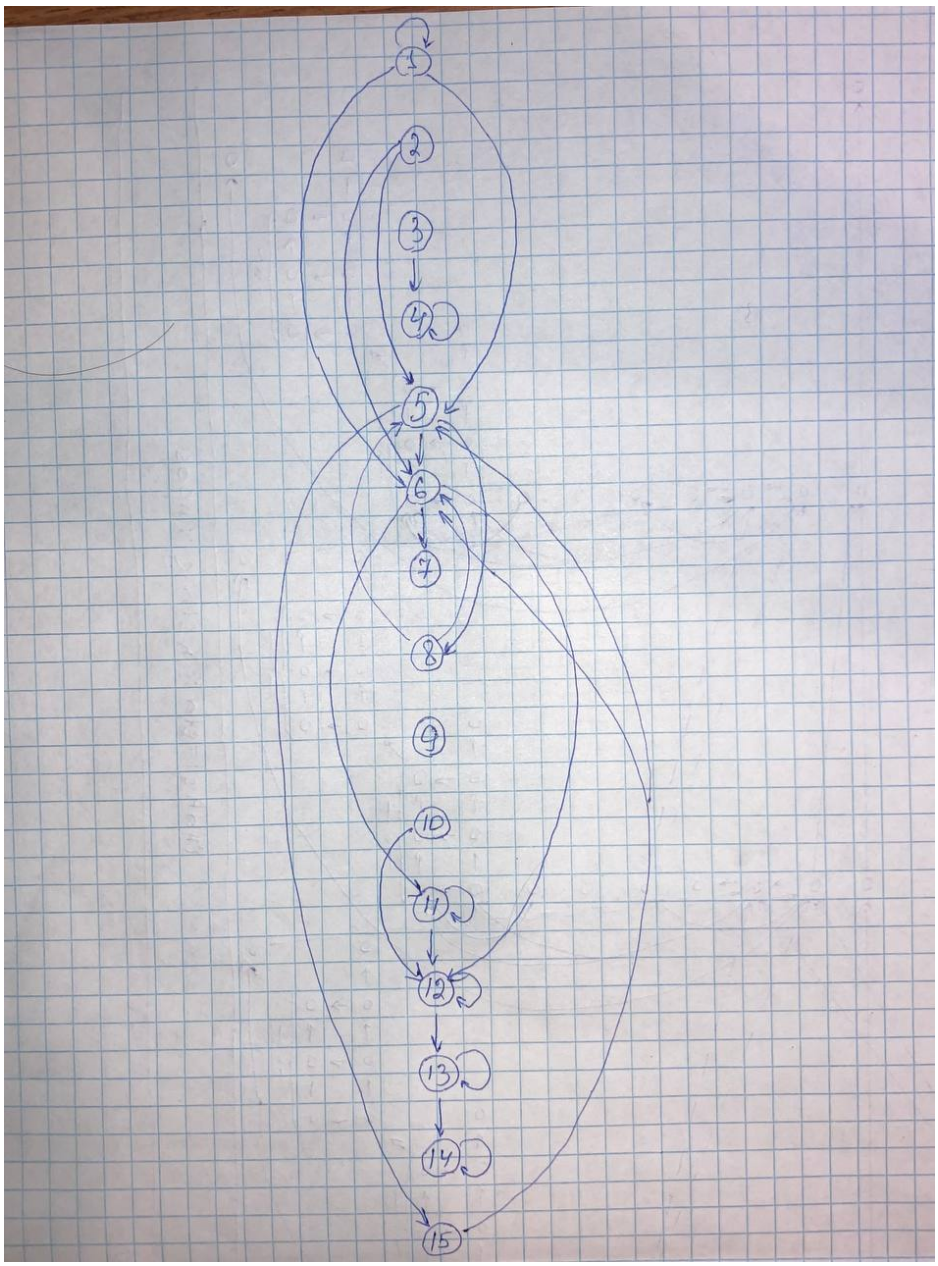

Рисунок 2.2 – Информационный граф

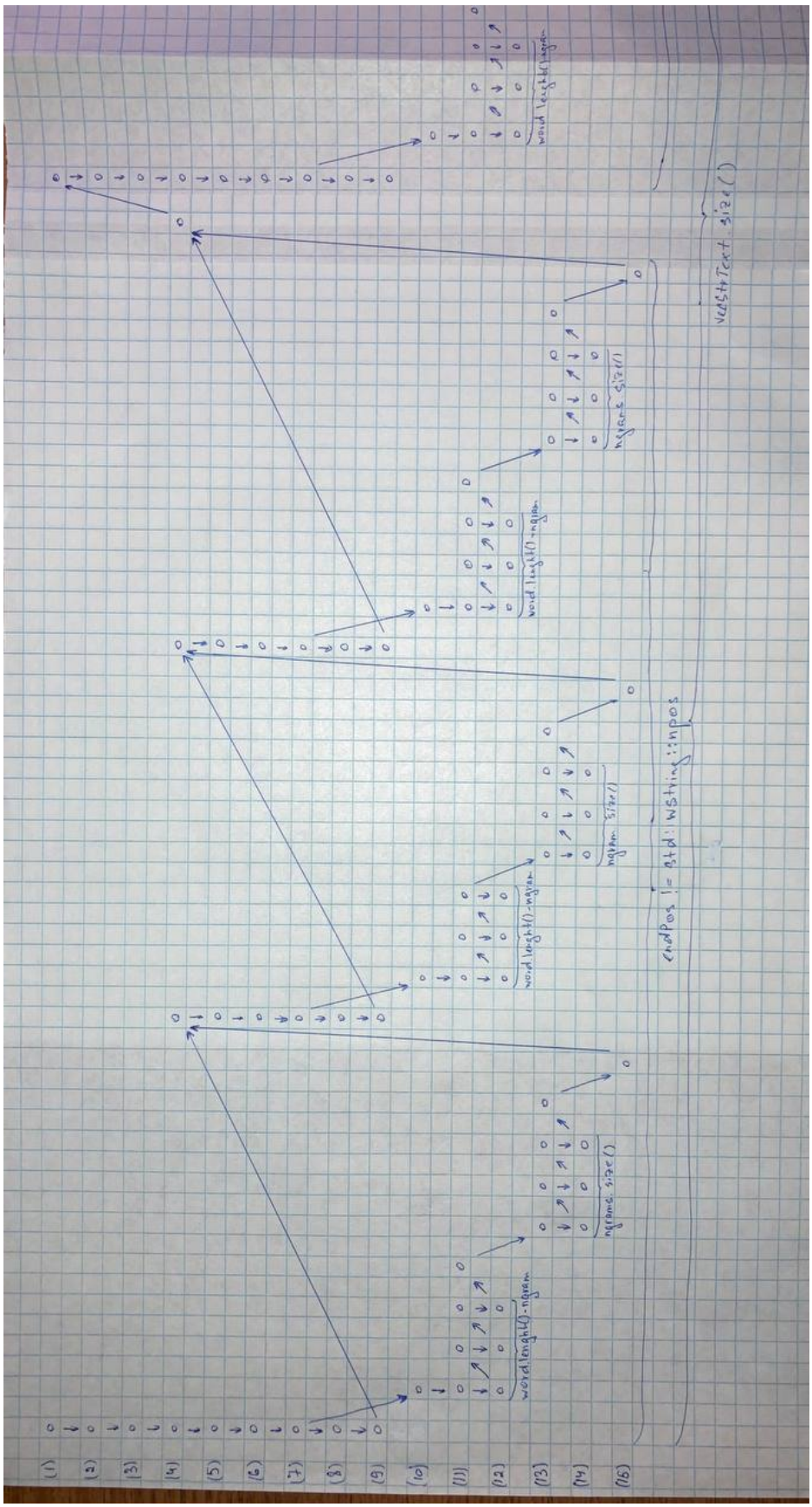

Рисунок 2.3 – Операционная история (начало)

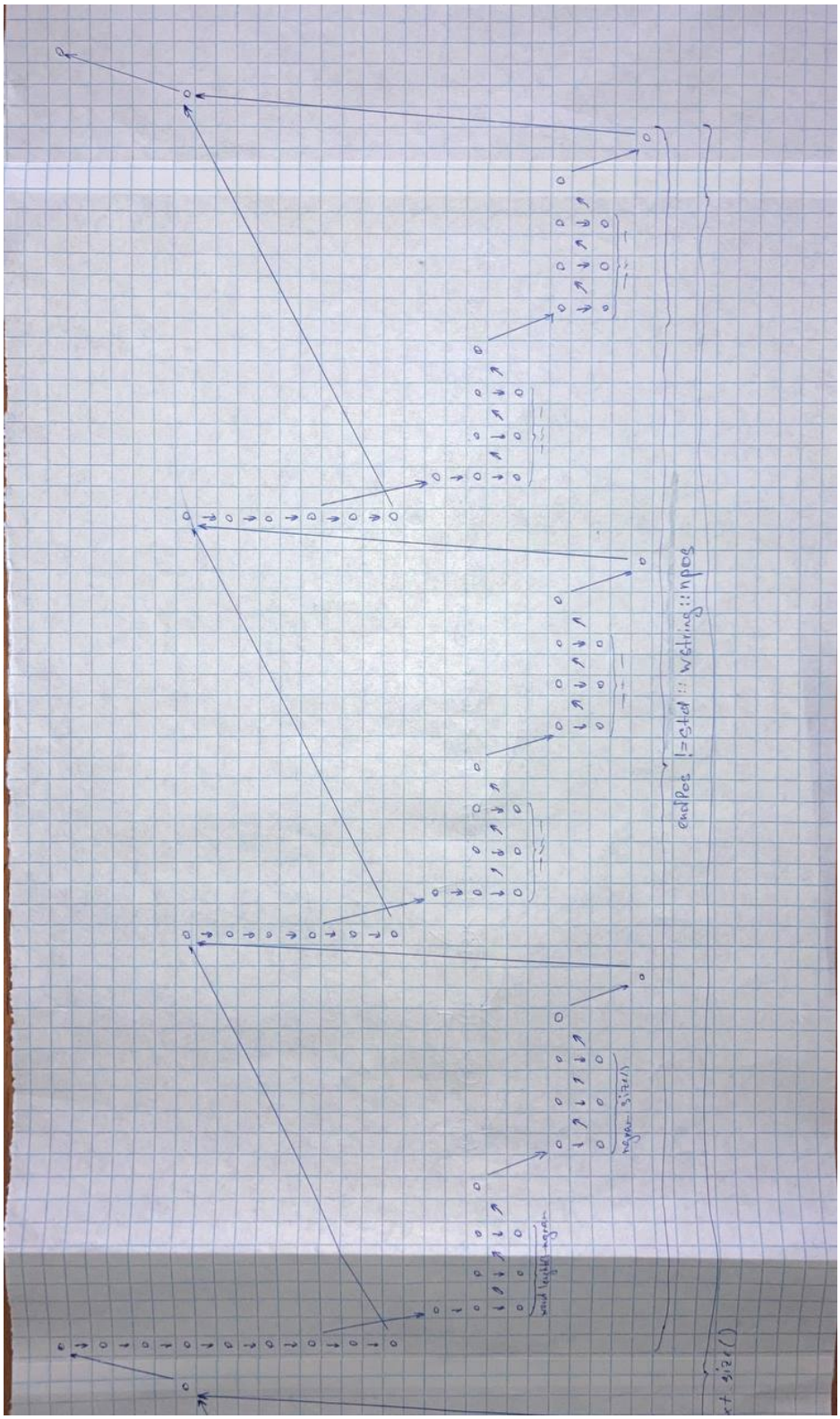

Рисунок 2.4 – Операционная история (конец)

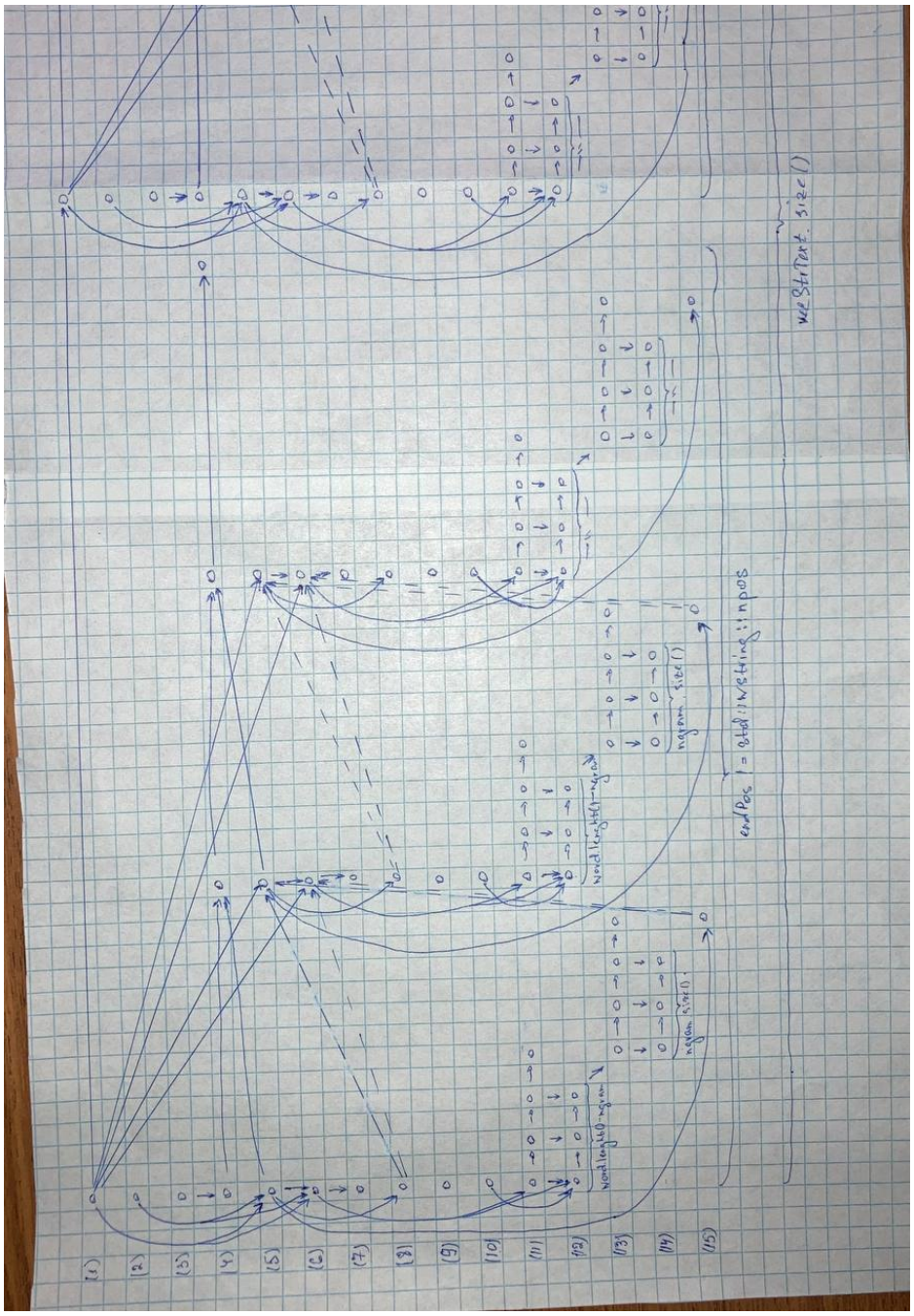

Рисунок 2.5 – Информационная история (начало)

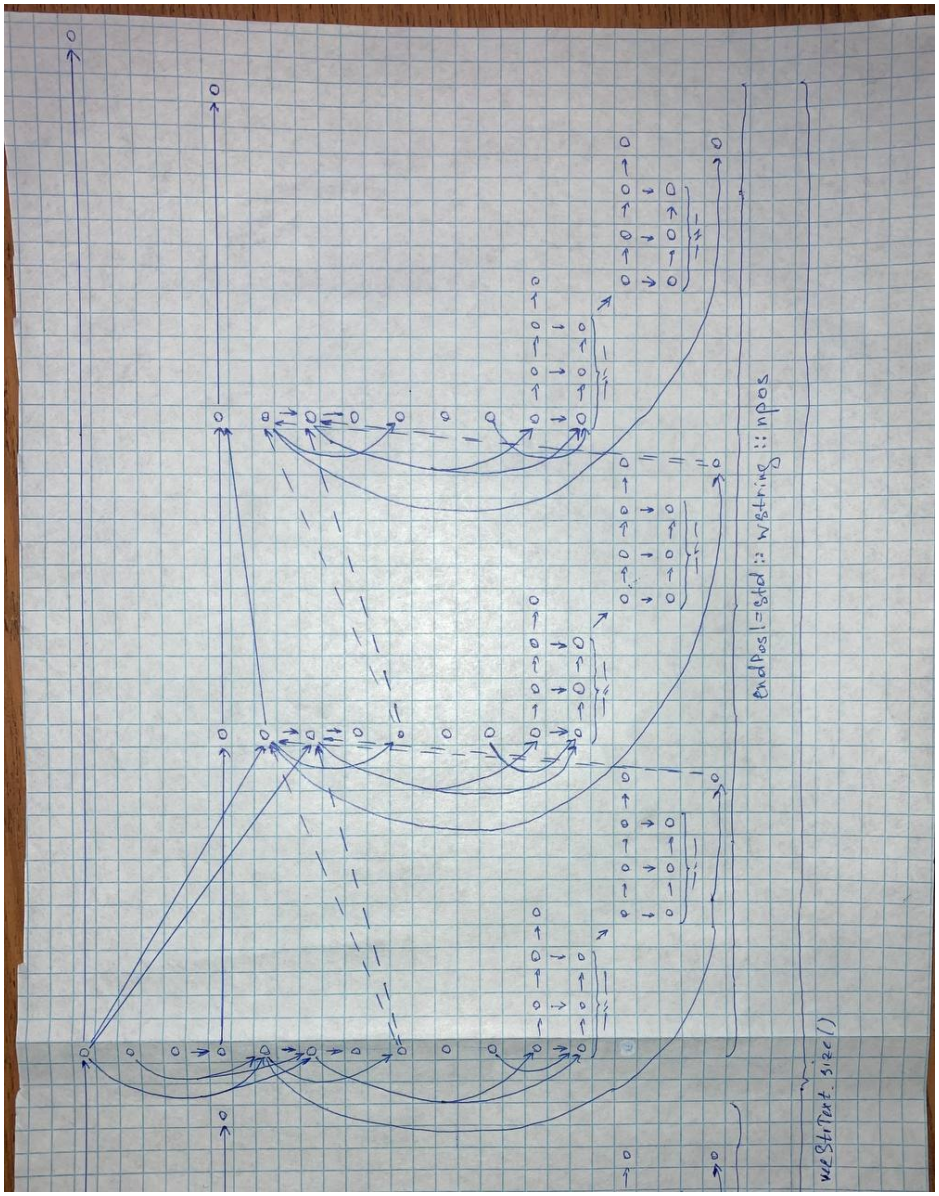

Рисунок 2.6 – Информационная история (конец)

2.4 Возможность распараллеливания

В качестве способа распараллеливания можно разделить строки файла между потоками и запустить обработку частей текста в отдельных потоках, а затем объединить результаты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Справочник по языку C++ [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/cpp/cpp/cpp-language-reference?view=msvc-170 (дата обращения: 28.09.2022).