**Национальный исследовательский университет**

**«Высшая школы экономики»**

**Факультет компьютерных наук**

**Департамент программной инженерии**

**Контрольное домашнее задание**

**по дисциплине «Алгоритмы и структуры данных»**

**Тема работы: Алгоритмы сжатия.**

Выполнил:

Студент группы БПИ184(2)

Шебаршин П. В.

10 апреля 2020 г.

E-mail адрес:[pvshebarshin@edu.hse.ru](mailto:pvshebarshin@edu.hse.ru)

Преподаватель:

Варгулёв А.С.

«10» апреля 2020 г.

Москва, 2020 год. Модуль 4

**Оглавление**

[1. Постановка задачи 3](#_Toc5969289)

[2. Описание алгоритмов и использованных структур данных 4](#_Toc5969290)

[3. Описание плана эксперимента 6](#_Toc5969291)

[4. Результаты экспериментов 7](#_Toc5969292)

[5. Сравнительный анализ алгоритмов 8](#_Toc5969293)

[6. Заключение 9](#_Toc5969294)

[7. Источники 10](#_Toc5969295)

# Постановка задачи

Работа над контрольным домашним заданием заключалась в выполнения следующих задач:

1. Реализовать архиватор на основе кода Шеннона-Фано, используя язык С++.
2. Реализовать архиватор на основе алгоритма LZ77, используя язык С++.

2.1 Размер скользящего окна 5 Кб, размер словаря 4 Кб.

2.2 Размер скользящего окна 10 Кб, размер словаря 8 Кб архивированный.

2.3 Размер скользящего окна 20 Кб, размер словаря 16 Кб архивированный.

1. Вычислить энтропию файлов, подсчитать время работы каждого алгоритма, вычислить коэффициенты сжатия.

# Описание алгоритмов и использованных структур данных

Для реализации нам было представлено 4 алгоритма: Шенона-Фано, LZ77, LZW, Хафмена.

2.1 **Алгоритм Шенона-Фано**:

1. Символы первичного алфавита m1 выписывают по убыванию вероятностей.
2. Символы полученного алфавита делят на две части, суммарные вероятности символов которых максимально близки друг другу.
3. В префиксном коде для первой части алфавита присваивается двоичная цифра «0», второй части — «1».
4. Полученные части рекурсивно делятся и их частям назначаются соответствующие двоичные цифры в префиксном коде.

2.2 **Алгоритм LZ77**:

Метод кодирования, согласно принципу скользящего окна, учитывает уже ранее встречавшуюся информацию, то есть информацию, которая уже известна для кодировщика и декодировщика (второе и последующие вхождения некоторой строки символов в сообщении заменяются ссылками на её первое вхождение).

Благодаря этому принципу алгоритмы LZ\* иногда называются методами сжатия с использованием скользящего окна. Скользящее окно можно представить в виде буфера (или более сложной динамической структуры данных), который организован так, чтобы запоминать «сказанную» ранее информацию и предоставлять к ней доступ. Таким образом, сам процесс сжимающего кодирования согласно LZ77 напоминает написание программы, команды которой позволяют обращаться к элементам «скользящего окна», и вместо значений сжимаемой последовательности вставлять ссылки на эти значения в «скользящем окне». Размер скользящего окна может динамически изменяться и составлять 2, 4 или 1 килобайта. Следует также отметить, что размер окна кодировщика может быть меньше или равен размеру окна декодировщика, но не наоборот.

2.3 **Алгоритм LZW**:

Примечательной особенностью алгоритма LZW является простота реализации, благодаря которой он до сих пор очень популярен, несмотря на зачастую худшую степень сжатия по сравнению с такими аналогами, как LZ77. Обычно LZW реализуется с помощью префиксного дерева, содержащего фразы из словаря: для нахождения W нужно просто прочитать как можно более длинную строку из корня дерева, затем для добавления новой фразы WK нужно присоединить к найденному узлу нового сына по символу K, а кодом фразы W может выступать индекс узла в массиве, содержащем все узлы.

Все алгоритмы разбиты по классам, что позволяет быстро обращаться к тому или иному алгоритму.

2.4 **Алгоритм Хафмена**:

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.
2. Выбираются два свободных узла дерева с наименьшими весами.
3. Создается их родитель с весом, равным их суммарному весу.
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.
5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0. Битовые значения ветвей, исходящих от корня, не зависят от весов потомков.
6. Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

В реализации алгоритмов были использованы структуры данных из стандартной библиотеки С++, такие как map, vector.

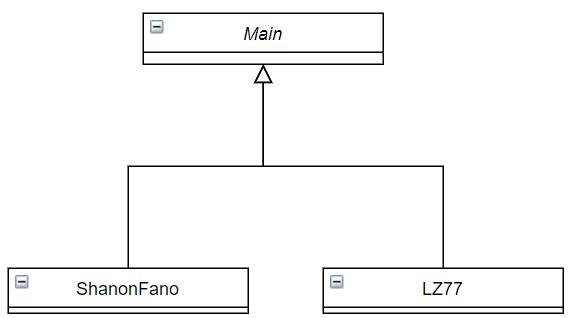
# Описание плана эксперимента

* 1. Реализованные алгоритмы планировалось запустить 10 раз на каждом из файлов для каждого из 3 алгоритмов, проводя при этом вычисления времени работы (с помощью std::chrono::steady\_clock::now()).
  2. Сохранять результаты каждого прохода по всем алгоритмам и файлам
  3. Вычислить по полученным результатам среднее время работы алгоритмов на каждом из файлов
  4. Визуализировать данные и проанализировать их

Результаты экспериментов записаны в файлы с названием input, однако алгоритм lz77 лежит в inputlz так как работал он мендленно.

# 4. Результаты экспериментов и диаграмма классов

Графики и таблица находятся в файле «Таблицы и графики.xlsx»



# 5. Сравнительный анализ алгоритмов

По всем приведенных данным в пункте 4 можно сделать некоторые выводы:

1. Полное время работы алгоритма Шенона-Фано быстрее чем время работы алгоритма LZ77, однако распаковка LZ77 очень быстрая.
2. Алгоритм Шенона-Фано лучше сжимает файлы с высокой энтропией.
3. Оба алгоритма могут увеличить размер файла.

Можно сделать вывод, что алгоритм Диница является самым быстрым из предложенных алгоритмов (менее всего зависит от количества ребер). А алгоритм Форда-Фалкерсона существенно проигрывает по времени работы.

# 6. Заключение

1. Изучил 4 алгоритма сжатия файлов.
2. Реализовал 2 алгоритма сжатия файлов с использованием языка C++.
3. Провел вычислительный эксперимент с целью оценки реализованных алгоритмов на 6 файлах разного типа.
4. Сравнила полученные данные, визуализировала их и сделала выводы.

# 7. Источники

1. <https://lms.hse.ru/>
2. <https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%A5%D0%B0%D1%84%D1%84%D0%BC%D0%B0%D0%BD%D0%B0>
3. <https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A8%D0%B5%D0%BD%D0%BD%D0%BE%D0%BD%D0%B0_%E2%80%94_%D0%A4%D0%B0%D0%BD%D0%BE>
4. <https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D1%8B_LZ77_%D0%B8_LZ78>
5. <https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_LZW>