|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | | |  |
|  | Институт информационных технологий (ИТ) | |
|  | Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ) | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**  **«Работа с данными из файла»** | | | |  |
| **по дисциплине «Структуры и алгоритмы обработки данных (часть 2/2)»** | | | |  |
|  | | | |  |
| Выполнил студент группы ИКБО-41-23 | | Попов А.В. | |  |
|  | |  | |  |
| Принял  *Ассистент* | | Рысин М.Л. | |  |
| Практические работы выполнены | «\_\_\_»\_\_\_\_\_\_\_2024 г. | | (подпись студента) | |
| «Зачтено» | «\_\_\_»\_\_\_\_\_\_\_2024 г. | | (подпись преподавателя) | |
|  |  | |  | |

СОДЕРЖАНИЕ

[ЦЕЛЬ РАБОТЫ 3](#__RefHeading___Toc1849_1074957294)

[ЗАДАНИЕ 1 4](#__RefHeading___Toc1851_1074957294)

[Задание 1.а 4](#__RefHeading___Toc1853_1074957294)

[Задание 1.б 5](#__RefHeading___Toc1855_1074957294)

[Задание 1.в 6](#__RefHeading___Toc1857_1074957294)

[ЗАДАНИЕ 2 7](#__RefHeading___Toc1859_1074957294)

[Задание 2.а 7](#__RefHeading___Toc1861_1074957294)

[Задание 2.б 8](#__RefHeading___Toc1863_1074957294)

[Задание 2.в 10](#__RefHeading___Toc1865_1074957294)

[ЗАДАНИЕ 3 12](#__RefHeading___Toc1867_1074957294)

[Задание 3.а 12](#__RefHeading___Toc1869_1074957294)

[Задание 3.б 15](#__RefHeading___Toc1871_1074957294)

[ВЫВОД 16](#__RefHeading___Toc1873_1074957294)

# ЦЕЛЬ РАБОТЫ

Освоить приёмы работы с битовым представлением беззнаковых целых чисел, реализовать эффективный алгоритм внешней сортировки на основе битового массива.

# ЗАДАНИЕ 1

## Задание 1.а

*Задача:*

Установить 5-й бит произвольного целого числа в 0.

*Описание алгоритма:*

Для выполнения данной задачи нужно создать маску, равную единице, после при помощи побитового перемещения сдвигаем всё на 4 позиций влево, инверсируем и побитово перемножаем результат на число x.

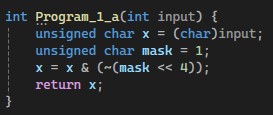
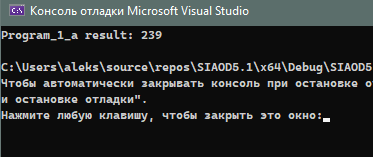
*Код программы:*

Рисунок 1 — Реализованный код для задачи 1.a

*Результат тестирования:*

Рисунок 2 — Вывод программы для входного значения «255»

## Задание 1.б

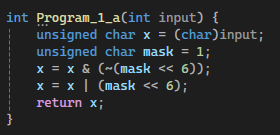
*Задача:*

Реализовать по аналогии с предыдущим примером установку 7- го бита числа в единицу.

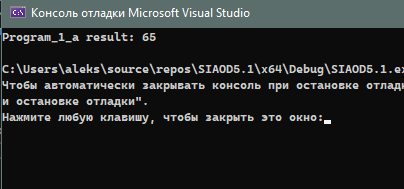
*Описание алгоритма:*

Для выполнения данной задачи нужно создать маску, равную единице, после при помощи побитового перемещения сдвигаем всё на 6 позиций влево, инверсируем и побитово перемножаем результат на число x. Таким образом мы гарантированно получаем 0 в 7-ом бите и далее прибавляем к числу маску в её исходном виде.

*Код программы:*

Рисунок 3 - Реализованный код для задачи 1.б

*Результат тестирования:*

Рисунок 4 — Вывод программы для входного значения «1»

## Задание 1.в

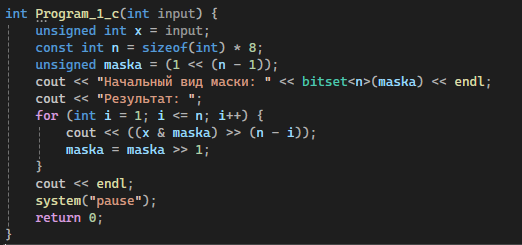
*Задача:*

Реализовать код листинга 1, объясните выводимый программой результат.

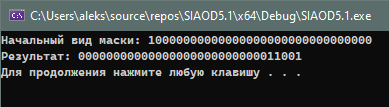
*Описание алгоритма:*

Алгоритм выводит побитовое представление числа x (в данном примере 25) в виде 32-битного числа. Для этого используется маска с единицей в старшем бите, которая последовательно сдвигается вправо, проверяя каждый бит числа x. Для каждого бита выводится его значение (0 или 1), начиная с самого старшего.

*Код программы:*

Рисунок 5 — Код листинга 1

*Результат тестирования:*

Рисунок 6 — Вывод программы для входного значения «25»

# ЗАДАНИЕ 2

## Задание 2.а

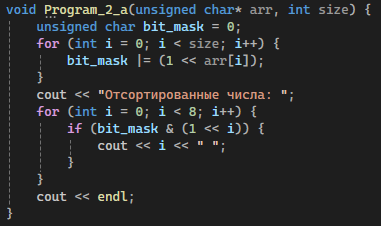
*Задача:*

Сортировка не более 8-и неповторяющихся чисел в диапазоне [0-7] типа данных unsigned char.

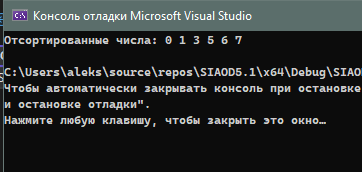
*Описание алгоритма:*

Алгоритм сортировки неповторяющихся чисел в диапазоне [0-7] с использованием битового массива основывается на использовании битовой маски. Изначально создается переменная типа unsigned char, которая будет служить битовой маской, где каждый бит соответствует одному числу из диапазона от 0 до 7. Для заполнения битовой маски перебираем все числа из входного массива. Если число присутствует, устанавливаем соответствующий бит в маске в 1 с помощью операции побитового сдвига и побитового ИЛИ. После заполнения битовой маски, чтобы получить отсортированные числа, последовательно проверяем каждый бит маски от 0 до 7. Если бит установлен в 1, выводим соответствующее число, используя операцию побитового И.

*Код программы:*

Рисунок 7 — Реализованный код задания 2.а

*Результат тестирования:*

Рисунок 8 — Вывод программы для входного массива «{ 5, 3, 7, 1, 6, 0 }»

## Задание 2.б

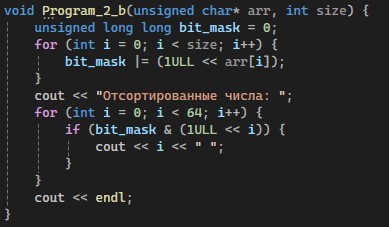
*Задача:*

Адаптировать вышеприведённый пример для набора из 64-х чисел (со значениями от 0 до 63) с битовым массивом в виде числа типа unsigned long long.

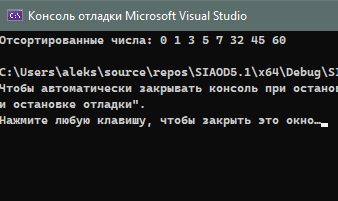
*Описание алгоритма:*

Для выполнения данного задания большая часть кода переходит с прошлого задания и меняются лишь некоторые переменные. Размер изменяется на 64, диапазон увеличивается до 63 и будет использовано 1ULL для правильной работы программы (ULL — это суффикс, который означает, что число будет представлено как unsigned long long).

*Код программы:*

Рисунок 9 - Реализованный код задания 2.б

*Результат тестирования:*

Рисунок 10 — Вывод программы для входного

массива «{ 5, 3, 60, 1, 45, 7, 32, 0 }»

## Задание 2.в

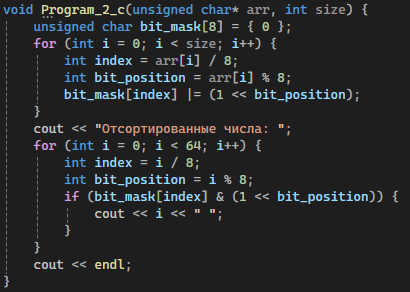
*Задача:*

Исправить программу задания 2.б, чтобы для сортировки набора из 64-х чисел использовалось не одно число типа unsigned long long, а линейный массив чисел типа unsigned char

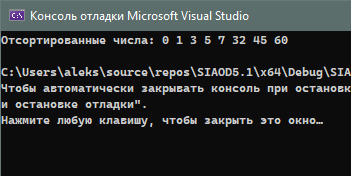
*Описание алгоритма:*

Программа выполняет сортировку чисел в диапазоне от 0 до 63 с использованием битового массива, представленного как линейный массив из 8 элементов типа unsigned char. Каждый элемент массива хранит 8 битов, что в сумме дает 64 бита, достаточные для представления всех чисел в диапазоне. Сначала программа инициализирует массив битовой маски, где все биты установлены в 0, затем для каждого числа из входного массива вычисляется байт, в котором оно должно находиться (делением числа на 8), и битовая позиция внутри этого байта (остатком от деления на 8). Соответствующий бит устанавливается в 1. После того как все числа обработаны, программа проходит по битовой маске и проверяет каждый бит. Если бит установлен в 1, программа выводит индекс этого бита.

*Код программы:*

Рисунок 11 - Реализованный код задания 2.в

*Результат тестирования:*

Рисунок 12 — Вывод программы для входного

массива «{ 7, 0, 45, 1, 60, 5, 32, 3 }»

# ЗАДАНИЕ 3

## Задание 3.а

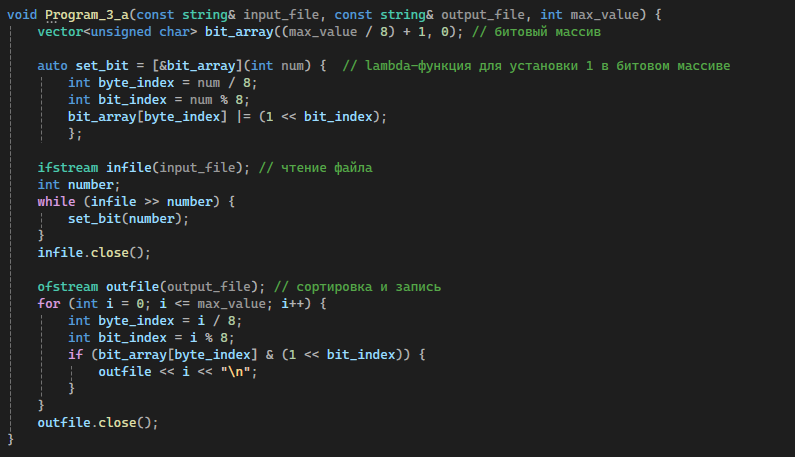
*Задача:*

Реализовать задачу сортировки числового файла, содержащего не более n=107 неотрицательных чисел, среди которых нет повторяющихся. Результатом должна быть упорядоченная по возрастанию последовательность исходных чисел в выходном файле.

*Описание алгоритма:*

Сначала определяется максимальное значение, которое составляет 107 - 1. На основе этого значения создается битовый массив, где каждый бит соответствует числу в данном диапазоне. Размер массива рассчитывается как ((максимальное значение / 8) + 1), что позволяет учесть все возможные числа, поскольку один байт может хранить 8 битов. Затем программа открывает входной файл с уникальными числами и считывает каждое число. Для каждого числа вызывается лямбда-функция, которая устанавливает соответствующий бит в битовом массиве. Установка бита осуществляется путем деления числа на 8 для определения индекса байта и взятия остатка от деления на 8 для определения индекса бита внутри байта. После того как все числа были прочитаны и биты установлены, программа открывает новый файл для записи отсортированных чисел. Она проходит по битовому массиву, проверяя каждый бит. Если бит установлен, программа записывает соответствующее число в выходной файл. В конце выполнения программы пользователю выводится сообщение о том, что файл был успешно отсортирован и записан, а также время, затраченное на выполнение операции.

*Код программы:*

Рисунок 13 - Реализованный код задания 3.а

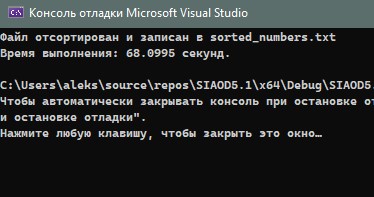
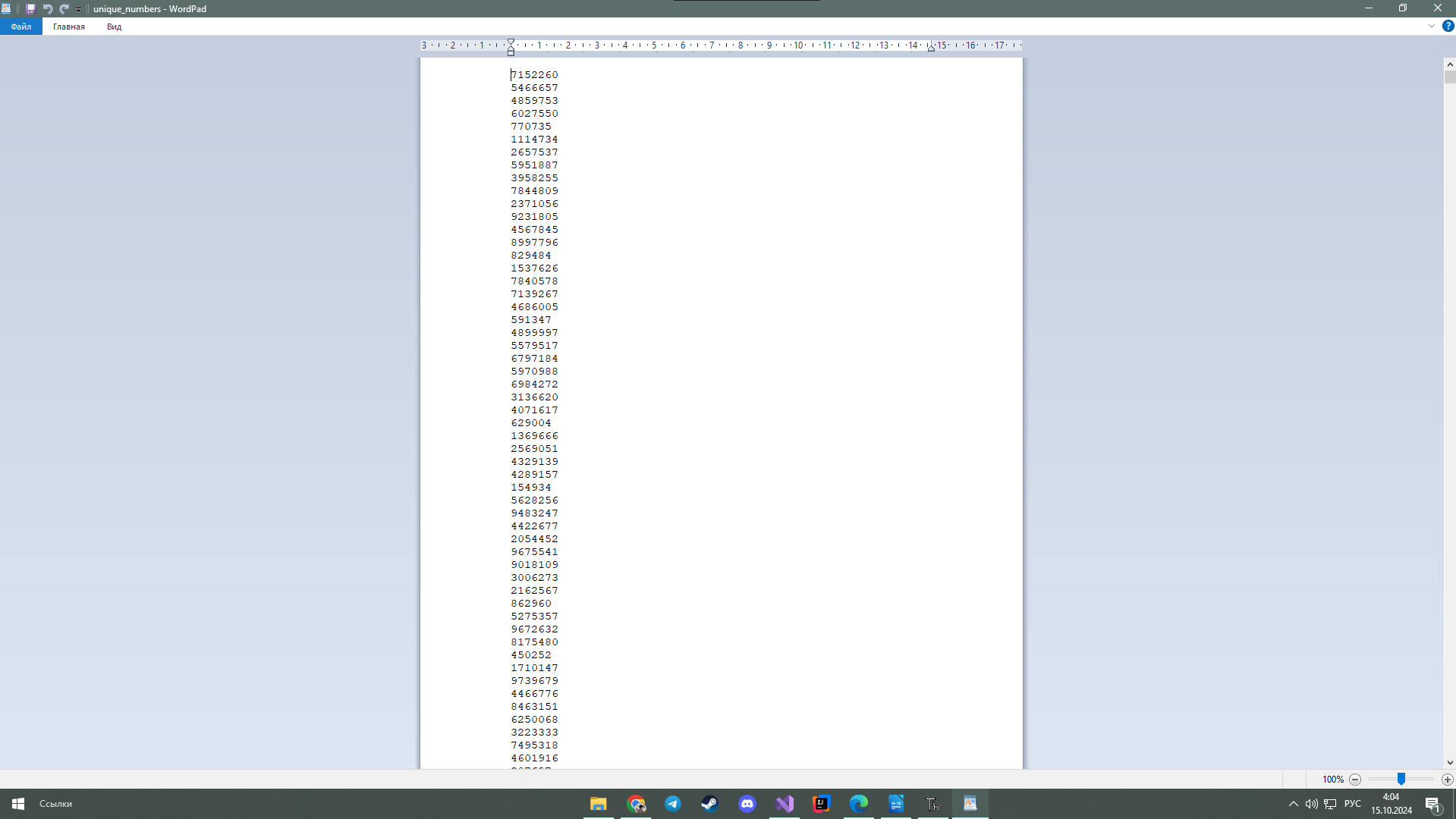
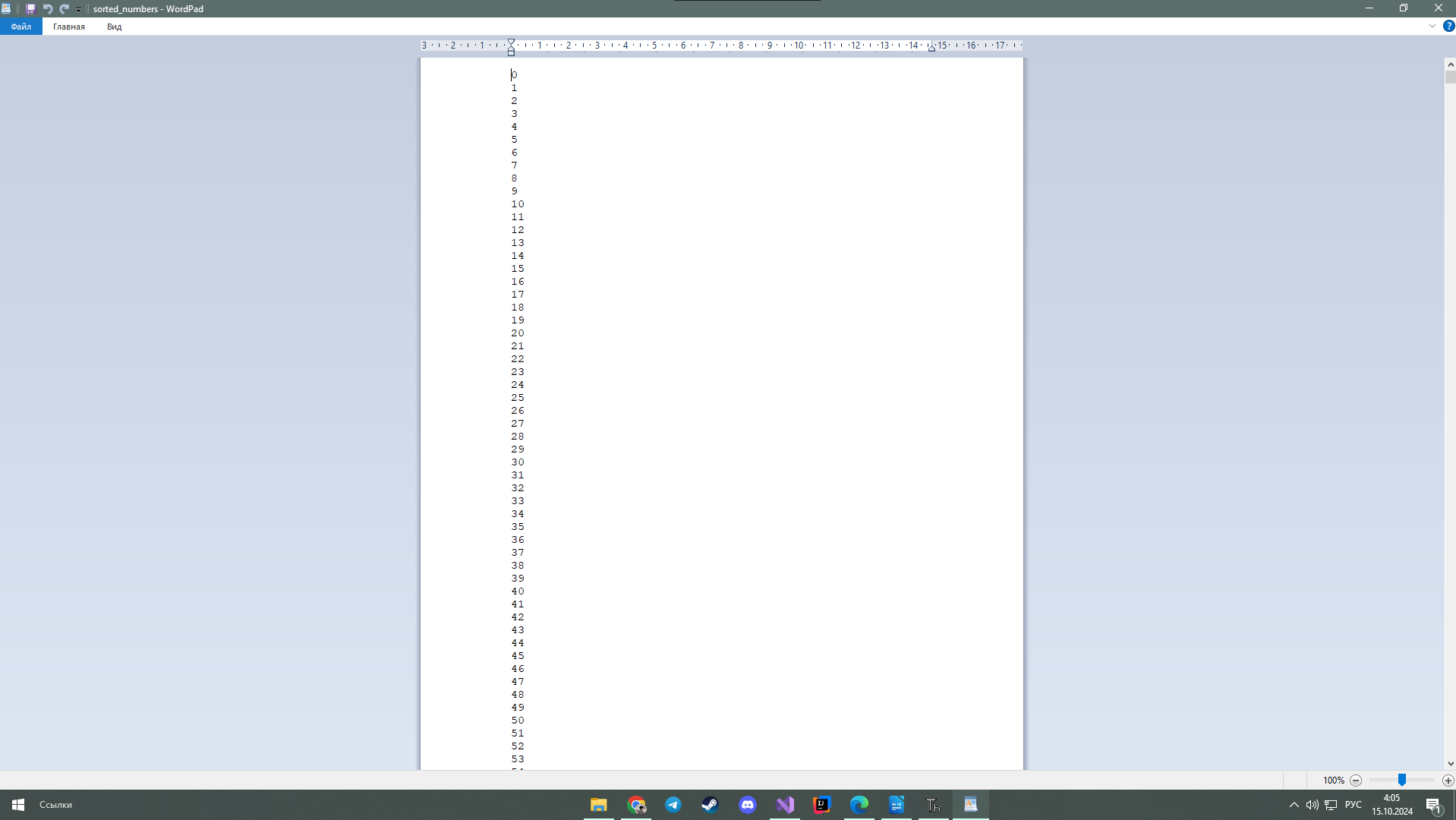
*Результат тестирования:*

Рисунок 14 - Вывод программы

Рисунок 15 — Неотсортированный файл

Рисунок 16 — Отсортированный файл

## Задание 3.б

*Задача:*

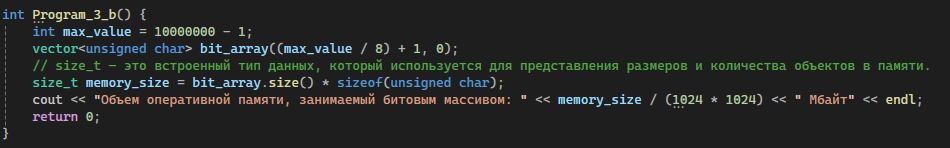
Определить программно объём оперативной памяти, занимаемый битовым массивом.

*Описание алгоритма:*

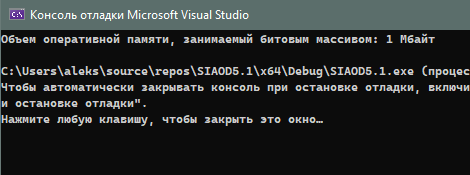
Объем памяти, занимаемый массивом, можно рассчитать по формуле:

Объем памяти (в байтах) = Размер массива × Размер одного элемента

*Код программы:*

Рисунок 17 - Реализованный код задания 3.б

*Результат тестирования:*

Рисунок 18 - Вывод программы

# ВЫВОД

Были освоены приёмы работы с битовым представлением беззнаковых целых чисел и реализован эффективный алгоритм внешней сортировки на основе битового массива.