

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

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)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ 5\_2**

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

Тема. Внешние структуры данных: текстовый и двоичный файлы.

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы ИКБО-60-23 |  | Шеенко В.А |
| Принял старший преподаватель |  | Скворцова Л.А. |

Москва 2024

СОДЕРЖАНИЕ

[1 ЦЕЛЬ РАБОТЫ 4](#_Toc177680851)

[2. ЗАДАНИЕ №1 5](#_Toc177680852)

[2.1 Постановка задания 5](#_Toc177680853)

[2.2 Задача 1 5](#_Toc177680854)

[2.2.1 Требования 5](#_Toc177680855)

[2.2.2 Ход решения 6](#_Toc177680856)

[2.2.3 Программная реализация 7](#_Toc177680857)

[2.3 Задача 2 8](#_Toc177680858)

[2.3.1 Требования 8](#_Toc177680859)

[2.3.2 Ход решения 9](#_Toc177680860)

[2.3.3 Программная реализация 10](#_Toc177680861)

[2.4 Задача 3 11](#_Toc177680862)

[2.4.1 Требования 11](#_Toc177680863)

[2.4.2 Ход решения 11](#_Toc177680864)

[2.4.3 Программная реализация 12](#_Toc177680865)

[2.5 Задача 4 12](#_Toc177680866)

[2.5.1 Требования 12](#_Toc177680867)

[2.5.2 Ход решения 13](#_Toc177680868)

[2.5.3 Программная реализация 13](#_Toc177680869)

[2.6 Задача 5 13](#_Toc177680870)

[2.6.1 Требования 13](#_Toc177680871)

[2.6.2 Ход решения 14](#_Toc177680872)

[2.6.3 Программная реализация 15](#_Toc177680873)

[3 ЗАДАНИЕ 2 16](#_Toc177680874)

[3.1 Постановка задачи 16](#_Toc177680875)

[3.2 Описание структуры для решения задачи 16](#_Toc177680876)

[3.3 Алгоритм решения 17](#_Toc177680877)

[3.4 Тестовый пример 17](#_Toc177680878)

[3.5 Код программы 17](#_Toc177680879)

[3.6 Результаты тестирования программы 18](#_Toc177680880)

[ВЫВОД 20](#_Toc177680881)

[СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 21](#_Toc177680882)

# 1 ЦЕЛЬ РАБОТЫ

Получить навыки применения файловых потоков языка С++ (или файлов языка Си) по управлению текстовым и двоичным файлами.

Задание 1. Разработать программу, управления текстовым файлом.

Задание 2. Разработать программу управление двоичными файлами с записями фиксированной длины

# 2. ЗАДАНИЕ №1

## 2.1 Постановка задания

Реализовать упражнения:

− по применению побитовых операций для изменения значений битов в ячейке оперативной памяти;

− по созданию маски для изменения значения ячейки;

− по созданию выражения, содержащего побитовые операции, для выполнения определенной операции над значением ячейки.

Требования к выполнению:

1. Разработать программу, которая продемонстрирует выполнение задач варианта. Каждая задача должна представлять фрагмент кода этой программы.

2. Подготовить тестовые примеры для всех задач варианта.

Индивидуальный вариант

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Задача 1  Обнулить биты. Номера битов. | Задача 2 Установить в 1 Номера битов | Задача 3 множитель | Задача 4 делитель | Задача 5 |
| 28. | 5-ый, 11-ый, 3-ий | Четыре старших бита | 32 | 32 | 9, 25, 28, 17 |

## 2.2 Задача 1

### 2.2.1 Требования

1. Определить переменную целого типа, значение которой должно быть изменено. Для каждой задачи определить отдельные переменные.

2. Разработать оператор, который установит заданные в задании биты исходного значения переменной в значение, определенное в задаче варианта, используя соответствующую маску и поразрядную операцию.

Реализовать два решения задачи.

Решение 1. Переменной, значение которой должно быть изменено, присвоить значение, используя константу в шестнадцатеричной системе счисления. Маска представлена переменной, инициализированная шестнадцатеричной константой.

Решение 2. Маска формируется выражением с использованием значения 1 и побитовых операций, выражение явно встроено в оператор операции.

3. Исходные данные и результаты выполнения задачи выводить на монитор по формату. Для решения 1 маска в выводимой таблице представлена константами в соответствии с тестовым примером. Для решения 2 одним выражением, расположенным в столбцах строки маски таблицы.

### 2.2.2 Ход решения

Решение 1

Рассмотрим, необходимые для решения задачи, переменные и выражения:

unsigned int value – начальное изменяемое решение

unsigned int mask = 0xfffffbeb – Маска для первого решения, представленная в виде константы в шестнадцатеричной системе счисления.

unsigned int mask = ~((1 << 10)|(1 << 4)|(1 << 2)) – Маска для второго решения, представленная в виде побитовых операций над 1;

unsigned int res – результат изменения начального значения с помощью масок.

Для обнуления определенных битов будет использоваться операция побитового И. (0 & 1 = 0 и 0 & 0 = 0)

Таблица 1 – Тестовый пример для первого решения задачи №1 (value & mask)

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 42002 | a412 | 00000000000000001010010000010010 |
| Маска (mask) | 4294966251 | fffffbeb | 11111111111111111111101111101011 |
| Результат (res) | 40962 | a002 | 00000000000000001010000000000010 |

Таблица 2 – Тестовый пример для второго решения задачи №1 (value & mask)

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 41250 | a122 | 00000000000000001010000100100010 |
| Маска (mask)  ~((1 << 10)|(1 << 4)|(1 << 2) | 4294966251 | fffffbeb | 11111111111111111111101111101011 |
| Результат (res) | 40962 | a002 | 00000000000000001010000100100010 |

### 2.2.3 Программная реализация

Реализуем данные решения на языке программирования c++ (рис. 1-2). Результат работы - рис. 3-4

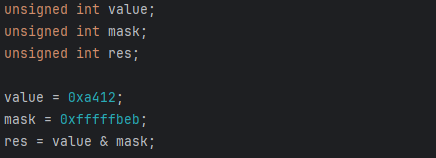


Рисунок 1 – Программная реализация 1-ого решения задачи №1

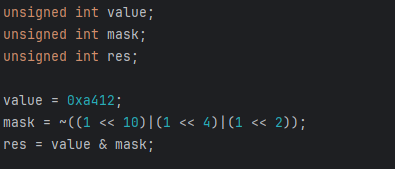


Рисунок 2 – Программная реализация 2-ого решения задачи №1

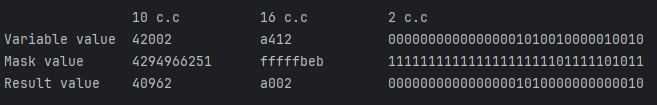


Рисунок 3 – Результат работы 1-ого решения

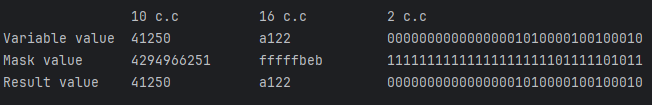


Рисунок 4 – Результат работы 2-ого решения

## 2.3 Задача 2

### 2.3.1 Требования

1. Определить переменную целого типа, значение которой должно быть изменено. Для каждой задачи определить отдельные переменные.

2. Разработать оператор, который установит заданные в задании биты исходного значения переменной в значение, определенное в задаче варианта, используя соответствующую маску и поразрядную операцию.

Реализовать два решения задачи.

Решение 1. Переменной, значение которой должно быть изменено, присвоить значение, используя константу в шестнадцатеричной системе счисления. Маска представлена переменной, инициализированная шестнадцатеричной константой.

Решение 2. Маска формируется выражением с использованием значения 1 и побитовых операций, выражение явно встроено в оператор операции.

3. Исходные данные и результаты выполнения задачи выводить на монитор по формату. Для решения 1 маска в выводимой таблице представлена константами в соответствии с тестовым примером. Для решения 2 одним выражением, расположенным в столбцах строки маски таблицы.

### 2.3.2 Ход решения

Используются все те же переменные, что и в первой задаче. Однако для установления единиц в битах необходимо использовать побитовую операцию ИЛИ (0 | 1 = 1 и 1 | 1 = 1 и 0 | 0 = 0)

Таблица 3 – Тестовый пример для первого решения задачи №2

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 47683 | ba43 | 00000000000000001011101001000011 |
| Маска (mask) | 4026531840 | f0000000 | 11110000000000000000000000000000 |
| Результат (res) | 4026579523 | f000ba43 | 11110000000000001011101001000011 |

Таблица 4 – Тестовый пример для второго решения задачи №2

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 53127 | cf87 | 00000000000000001100111110000111 |
| Маска (mask)  (1 << 31)|(1 << 30)|(1 << 29)|(1 << 28) | 4026531840 | f0000000 | 11110000000000000000000000000000 |
| Результат (res) | 4026584967 | f000cf87 | 11110000000000001100111110000111 |

### 2.3.3 Программная реализация

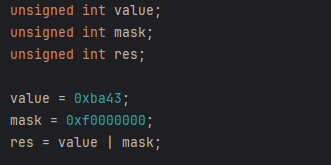


Рисунок 5 - Программная реализация 1-ого решения задачи №2

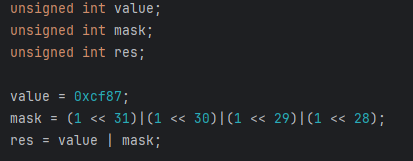


Рисунок6 - Программная реализация 2-ого решения задачи №2

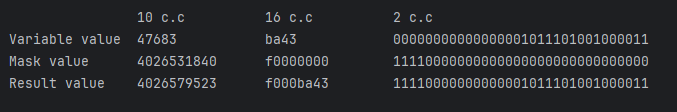


Рисунок 7 – Результат работы первого решения

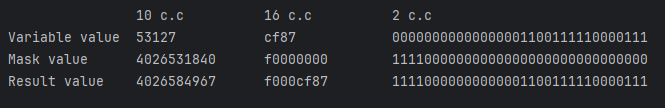


Рисунок 8 – Результат работы второго решения

## 2.4 Задача 3

### 2.4.1 Требования

Определить переменную целого типа. Разработать оператор присваивания и выражение, которое умножает значение переменной на число, указанное в задаче 3 варианта, используя соответствующую побитовую операцию. Изменяемое число вводится с клавиатуры.

Вывод тестового примера по формату табл. 1 с соответствующей модификацией: в строке Маска указываем только значение правого операнда операции в 10 с.с. и слово Маска заменяем на значение, соответствующего множителя.

### 2.4.2 Ход решения

По заданию необходимо произвольное число умножить на 32, что является степенью двойки , следовательно, необходимо сделать сдвиг битов начального числа на 5 влево.

Таблица 5 - Тестовый пример для задачи №3

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 4325 | 10e5 | 00000000000000000001000011100101 |
| Сдвиг влево | 5 |  |  |
| Результат (res) | 138400 | 21ca0 | 00000000000000100001110010100000 |

### 2.4.3 Программная реализация

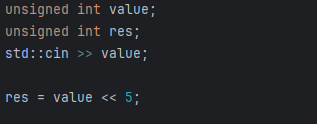


Рисунок 9 – Программная реализация задачи №3

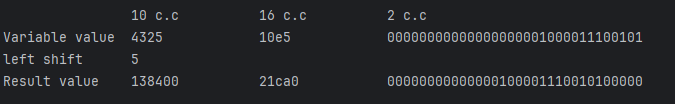


Рисунок 10 – Результат работы решения задачи №3

## 2.5 Задача 4

### 2.5.1 Требования

Определить переменную целого типа. Разработать оператор присваивания и выражение, которое делит значение переменной на число, указанное в задаче 4 варианта, используя соответствующую поразрядную операцию. Изменяемое число вводится с клавиатуры.

Вывод тестового примера по формату с соответствующей модификацией: в строке Маска указываем только значение правого операнда операции в 10 с.с. и слово Маска заменяем на значение, соответствующего делителя.

### 2.5.2 Ход решения

По аналогии с 3-ей задачей про умножение, однако для деления необходимо сдвинуть биты вправо.

Таблица 5 - Тестовый пример для задачи №4

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 2048 | 800 | 00000000000000000000100000000000 |
| Сдвиг вправо >> | 5 |  |  |
| Результат (res) | 64 | 40 | 00000000000000000000000001000000 |

### 2.5.3 Программная реализация

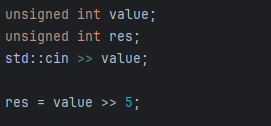


Рисунок 11 - Программная реализация задачи №4

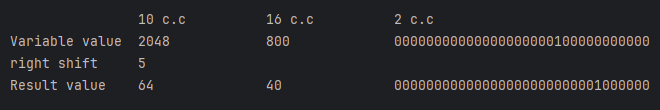


Рисунок 11 – Результат работы решения задачи №4

## 2.6 Задача 5

### 2.6.1 Требования

Определить переменную целого типа. Разработать оператор присваивания и выражение, в котором используются только поразрядные операции. В выражении используется маска – переменная. Маска может быть инициализирована единицей в младшем разряде (для нечетных вариантов) или единицей в старшем разряде (для четных вариантов). Изменяемое число вводится с клавиатуры.

### 2.6.2 Ход решения

Для инвертирования битов будет использовано побитовое исключающее ИЛИ (XOR). Переменные:

1. unsigned int value – начальное изменяемое значение
2. unsigned int mask – маска
3. unsigned int res – результат инвертирования определенных бит в value (при помощи маски)

Выражение для маски:

mask = 0x80000000 (1 в старшем разряде, согласно варианту)

mask = (mask >> 4)|(mask >> 7)|(mask >> 15)|(mask >> 23)

Таблица 6 - Тестовый пример для задачи №5

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 с.с. | 16 с.с. | 2 с.с. |
| Изменяемое значение (value) | 4400483 | 432563 | 00000000010000110010010101100011 |
| Маска (mask) | 151060736 | 9010100 | 00001001000000010000000100000000 |
| Результат (res) | 155329635 | 9422463 | 00001001010000100010010001100011 |

### 2.6.3 Программная реализация

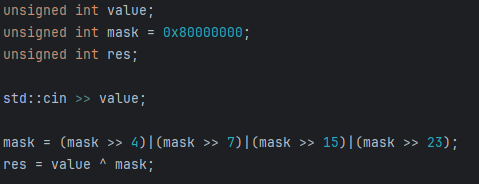


Рисунок 12 ­- Программная реализация задачи №5

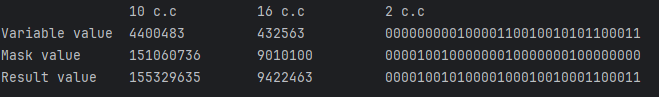


Рисунок 13 – Результат работы решения задачи №5

# 3 ЗАДАНИЕ 2

## 3.1 Постановка задачи

Дано: Файл, содержащий не более n (n=107) целых положительных

чисел, каждое из которых семизначное число, т.е. принадлежит диапазону [1000000..9999999] и среди них нет повторяющихся.

Результат: упорядоченный по возрастанию список чисел, сохраненных в файле.

Ограничения:

- Доступной оперативной памяти 1МБ.

- Дисковая память неограниченна.

- Время выполнения программы 10 секунд.

## 3.2 Описание структуры для решения задачи

В данном решении будет использоваться динамический массив (vector), содержащий данные типа char, для которого отводится 8 бит в памяти. Каждый бит отвечает за информацию о наличии конкретного числа, если значение бита равняется 1, то это говорит о то, что число равное произведению индекса в массиве и 8, увеличенному на индекс внутри типа char, существует. Размер массива будет зависеть от максимально возможного числа в неотсортированном файле (или исходном массиве).

Таблица 7 – Пример первых двух элементов массива

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | | | | | | | | 1 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |

В первой строке табл. 7 указаны индексы динамического массива, внутри которого помещены элемента типа char (8 бит, 2-ая строчка индексы от 0 до 7). В 3-ей строке указаны значения бит (1 – существует число из 4-ой строки, 0 – не существует).

## Алгоритм решения

При использовании битового массива для представления сортируемых чисел, программу можно представить, как последовательность из трех задач:

1. Инициализация бит массива нулевыми значениями;
2. Считывание целых чисел из файла и установка в 1 соответствующих бит;
3. Формирование упорядоченного выходного файла путем последовательной проверки бит массива.

## 3.4 Тестовый пример

Имеется исходный массив, состоящий из последовательности [3, 6, 2, 8, 23, 12, 20, 1, 0, 4, 11, 13]. Для того чтобы отсортировать массив необходимо сделать проход по массиву и записать в битовый массив bit\_array на нужных местах единицы. Таким образом мы получим следующий массив:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | | | | | | | | 1 | | | | | | | | … |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | … |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | … |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | … |

Наконец, необходимо пройтись по массиву bit\_array и выписать те значения, в номере бита которого 1.

## 3.5 Код программы

Реализуем данное решение на языке программирования C++ (рис. 14).

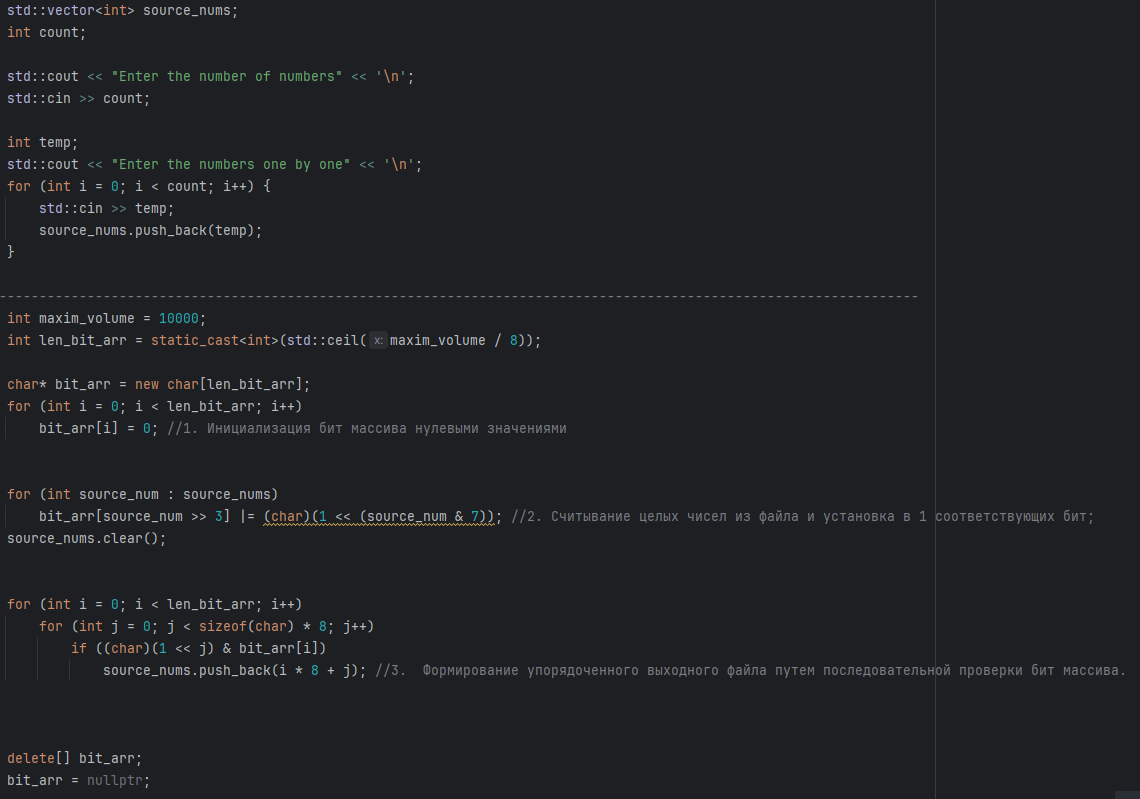


Рисунок 14 – Код программы сортировки из задания №2

## 3.6 Результаты тестирования программы



Рисунок 15 – Результат работы алгоритма на тестовом примере

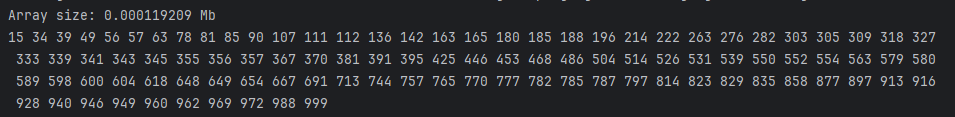


Рисунок 16 – Результат работы алгоритма для выборки из 100 чисел

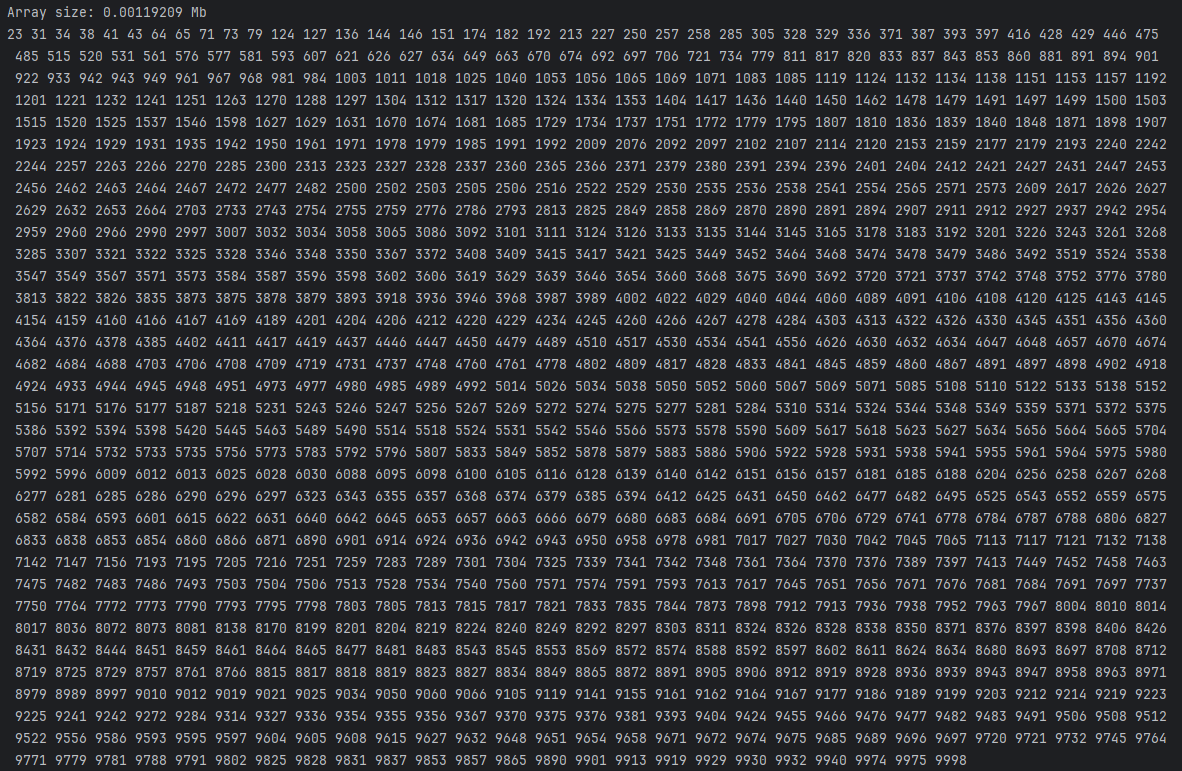


Рисунок 17 - Результат работы алгоритма для выборки из 1

# ВЫВОД

В результате выполнения данной практической работы были освоены навыки работы с побитовыми операциями, а также реализован на языке С++ эффективный алгоритм сортировки на основе битового массива.

# СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Рысин, М. Л. Введение в структуры и алгоритмы обработки данных : учебное пособие / М. Л. Рысин, М. В. Сартаков, М. Б. Туманова. — Москва : РТУ МИРЭА, 2022 — Часть 2 : Поиск в тексте. Нелинейные структуры данных. Кодирование информации. Алгоритмические стратегии — 2022. — 111 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/310826 (дата обращения: 10.09.2024).
2. ГОСТ 19.701-90. Единая система программной документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения : межгосударственный стандарт : дата введения 1992-01- 01 / Федеральное агентство по техническому регулированию. – Изд. официальное. – Москва : Стандартинформ, 2010. – 23 с