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 Толстов А.Э.

Принял Рысин М.Л.
Ассистент

Практические работы выполнены «___»_____2024 г. (подпись студента)

«Зачтено» «___»_____2024 г. (подпись преподавателя)

Москва 2024

Цель работы

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

Задание 1

Задание 1.A

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 позиции влево, инверсируем и перемножаем результат на число x.

Код программы:

```
int P1::Task1(int input)
{
    unsigned char x = (char)input;
    unsigned char mask = 1;
    x = x & (~(mask << 4));
    return x;
}
```

Рис. 1 Реализованный код для задачи 1.A

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    std::cout << Program1.Task1(255);

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

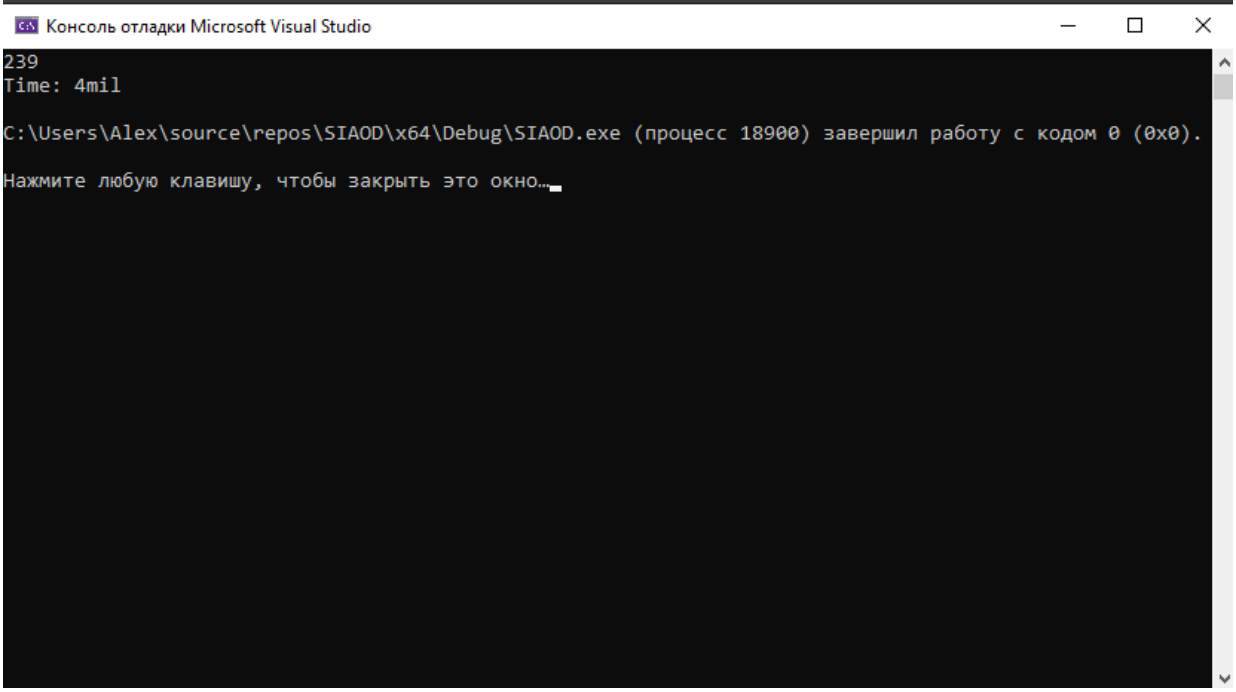

Рис. 2 Драйвер и вывод задания 1.A

Задание 1.Б

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

Описание алгоритма:

Для выполнения поставленной задачи нужно прочитать число и перенести его в char (8-битный формат хранения), объявить маску и инициализировать её значением единицы, после сдвинуть маску на 6 битов влево и применить инверсию маски к числу x при помощи побитового умножения и сложения.

Код программы:

```
int P1::Task2(int input)
{
    unsigned char x = input;
    unsigned char mask = 1;
    mask = mask << 6;
    x = x & (~(mask));
    x = x | mask;
    return x;
}
```

Рис. 3 Реализованный код для задачи 1.Б

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    std::cout << Program1.Task2(10);

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

```
Консоль отладки Microsoft Visual Studio
Time: 0mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 19744) завершил работу с кодом 0 (0x0).

Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 4 Драйвер и вывод задания 1.Б

Задание 1.В

Реализовать код листинга 1, объясните выводимый программой результат.

Описание алгоритма:

Алгоритм объявляет целочисленное число n и выделяет ему 8 ячеек памяти равные целочисленному числу, после создаётся маска, равная результату сдвига единицы на n – 1 позиций влево (в данном случае n – 1 равен 31). Программа выводит bitset маски, который равен единице и 31 нулю после него. Вывод числа с маской происходит в цикле for. Стоит заметить, как только целочисленные числа начиная от 2 147 483 648 будут меняться в значении. При вводе 25 в программу, вывод будет 27 нулей слева и 11001. Данное число из двоичной системы счисления в десятичную равен 25.

Код программы:

```
void P1::Task3(int input)
{
    const int n = sizeof(int) * 8;
    unsigned mask = (1 << n - 1);
    std::cout << "Starting mask value: " << std::bitset<n>(mask) << "\nResult: ";
    for (int i = 1; i <= n; i++)
    {
        std::cout << ((input & mask) >> (n - i));
        mask = mask >> 1;
    }
    std::cout << '\n';
}
```

Рис. 5 Код листинга 1 задачи 1.B

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program1.Task3(25);

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

Консоль отладки Microsoft Visual Studio

```
Starting mask value: 10000000000000000000000000000000
Result: 00000000000000000000000000011001

Time: 1mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 18688) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 6 Драйвер и вывод задания 1.B

Задание 2

Задание 2.A

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

Описание алгоритма:

Данное задание требует сортировку при помощи битового массива. Сначала объявляются переменные размера массива для ввода чисел, буфер числа, битовый массив типа данных char (8-битный массив данных). Первый цикл for является вводом чисел, где числа не могут быть больше 7 и не могут быть меньше 0. Если число в диапазоне, то в битовый массив меняется число 0 на число 1 в позиции числа, иначе выводит ошибку о несоблюдении инструкции ввода. Сортировка происходит во время заполнения чисел. После заполнения чисел вывести bitset (массив битов) и вывести отсортированный массив чисел, при помощи второго цикла for, в котором находится проверка, является ли текущая позиция единицей в битовом массиве.

Код программы:

```
void P2::Task1()
{
    const int SIZE = 8;
    int num;
    unsigned char bitArr = 0;
    // ввод значений
    std::cout << "Enter values from 0 to 7.\n\nMax amount of numbers is 8.\nEnter '9' to stop.\n";
    for (int i = 0; i < SIZE; i++)
    {
        std::cout << i + 1 << " number is... ";
        std::cin >> num;

        if (num == 9) break;

        if (num >= 0 && num < 8)
        {
            bitArr |= (1 << num);
        }
        else
        {
            std::cout << "Instruction not followed";
            i--;
        }
        std::cout << '\n';
    }
    std::cout << '\n' << std::bitset<SIZE>(bitArr) << '\n';
    for (int i = 0; i < SIZE; i++)
    {
        if (bitArr & (1 << i))
        {
            std::cout << i << " ";
        }
    }
}
```

Рис. 7 Реализация кода для задания 2.A

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program2.Task1();

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

Консоль отладки Microsoft Visual Studio

```
Enter values from 0 to 7.

Max amount of numbers is 8.
Enter '9' to stop.
1 number is... 3

2 number is... 2

3 number is... 1

4 number is... 5

5 number is... 9

00101110
1 2 3 5
Time: 6377mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 4264) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 8 Драйвер и вывод задания 2.A

Задание 2.Б

Сортировка не более 64-ёх неповторяющихся чисел в диапазоне [0-63] типа данных unsigned long long.

Описание алгоритма:

Для выполнения данного задания большинство кода идёт с прошлого задания и меняются некоторые переменные. Размер переходит на 64, диапазон до 63 и вместо обычной смены бита при помощи единицы и позиции числа будет использовано 1ULL (1 unsigned long long), для правильной работы программы. Остальное не отличается от кода задания 2.А

Код программы:

```
void P2::Task2()
{
    const int SIZE = 64;
    int num;
    unsigned long long bitArr = 0;
    // ввод значений
    std::cout << "Enter values from 0 to 63.\n\nMax amount of numbers is 64.\nEnter '65' to stop.\n";

    for (int i = 0; i < SIZE; i++)
    {
        std::cout << i + 1 << " number is... ";
        std::cin >> num;

        if (num == 65) break;

        if (num >= 0 && num < 64)
        {
            bitArr |= (1ULL << num);
        }
        else
        {
            std::cout << "Instruction not followed";
            i--;
        }
        std::cout << '\n';
    }

    std::cout << '\n' << std::bitset<SIZE>(bitArr) << '\n';

    for (int i = 0; i < SIZE; i++)
    {
        if (bitArr & (1ULL << i))
        {
            std::cout << i << " ";
        }
    }
}
```

Рис. 9 Реализация кода для задания 2.Б

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program2.Task2();

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

Консоль отладки Microsoft Visual Studio

```
Enter values from 0 to 63.

Max amount of numbers is 64.
Enter '65' to stop.
1 number is... 32

2 number is... 31

3 number is... 30

4 number is... 1

5 number is... 65

00000000000000000000000000000001111000000000000000000000000010
1 30 31 32
Time: 5593mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 20464) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 10 Драйвер и вывод задания 2.Б

Задание 2.B

Сортировка не более 64-ёх неповторяющихся чисел в диапазоне [0-63] типа данных unsigned char.

Описание алгоритма:

Данное задание требует сортировку 64 чисел в 8-и битном массиве. Количество 8-и битных массивов не указано, следовательно объявляем количество массивов равным максимальному количеству чисел делённое на 8 (размер массива). После инициализируем массивы нулями. В первом цикле for изменяется подход к заполнению битовых массивов, где мы сначала узнаём в какой массив нужно положить число (числа от 0 по 7 идут в массив номера 0, 8 до 15 в 1, и так до номера 7, куда идут числа 56-63) и после получения данного числа, заносим единицу в позицию массива. Не забываем, что число может быть больше 8 и для точного переноса оставляем только остаток от числа. Вывод битового массива происходит во втором цикле for, идущий от обратного. Вывод отсортированного массива идёт при помощи двух циклов for, где первый определяет битовый массив, а второй определяет позицию в текущем массиве.

Код программы:

```
void P2::Task3()
{
    const int SIZE = 64;
    int num;
    unsigned char bitArr[SIZE / 8]{};
    for (int i = 0; i < 8; i++) bitArr[i] = int(0); // инициализация
    int temp = 0;
    // ввод значений
    std::cout << "Enter values from 0 to 63.\n\nMax amount of numbers is 64.\nEnter '65' to stop.\n";

    for (int i = 0; i < SIZE; i++)
    {
        std::cout << i + 1 << " number is... ";
        std::cin >> num;

        if (num == 65) break;

        if (num >= 0 && num < 64)
        {
            temp = num / 8;
            bitArr[temp] |= (1 << num % 8);
        }
        else
        {
            std::cout << "Instruction not followed";
            i--;
        }
        std::cout << '\n';
    }

    std::cout << '\n';
    for (int i = SIZE / 8 - 1; i >= 0; i--) std::cout << std::bitset<SIZE / 8>(bitArr[i]) << ' ';
    std::cout << '\n';

    for (int i = 0; i < SIZE / 8; i++)
    {
        for (int y = 0; y < 8; y++)
        {
            if (bitArr[i] & (1 << y))
            {
                std::cout << y + i * 8 << " ";
            }
        }
    }
}
```

Рис. 11 Реализация кода для задания 2.B

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program2.Task3();

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

```
Консоль отладки Microsoft Visual Studio
Enter values from 0 to 63.

Max amount of numbers is 64.
Enter '65' to stop.
1 number is... 1

2 number is... 3

3 number is... 5

4 number is... 62

5 number is... 65

01000000 00000000 00000000 00000000 00000000 00000000 00000000 00101010
1 3 5 62
Time: 4703mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 6568) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 12 Драйвер и вывод задания 2.B

Задание 3

Задание 3.A

Быстрая сортировка числового файла, содержащего не более 10000000 положительных неповторяющихся целых чисел, с помощью битового массива.

Описание алгоритма:

Данное задание подразумевает существование файла. Для этого создадим метод для создания файла и его заполнения

```
void P3::SpamFile()
{
    int _SIZE = 10000000;
    std::ofstream o_data("C:\\Users\\Alex\\source\\repos\\SIAOD\\Data.txt");
    for (int i = 0; i < _SIZE; i++)
    {
        o_data << rand() << '\n';
    }
}
```

Рис. 13 Реализация кода для создания файла с 1000000 чисел

Данный код создаёт файл с любым заданным размером. Числа идут от 0 до 32767, так как RAND_MAX в функции rand() равен 32767.

Основной код задания 3.A заключается в открытии файла, считывания данных из файла и записывания отсортированных чисел в файл. Объявляем переменные для ввода и вывода данных из файла, буфер максимальной размерности, целочисленное число размера всех чисел, число для определения позиции в битовом массиве и битовый вектор для хранения данных. Размер вектора определяется максимальным размером данных делённое на 64 (размер ULL), после заполняем весь вектор нулями. Первый цикл while считывает данные из файла в буфер. Внутри цикла while сначала считается позиция на битовом поле при помощи деления значения буфера на 64, после меняется текущая позиция в битовом массиве на единицу. Запись обратно в файл происходит при помощи двух циклов for, где первый обозначает какой вектор

используется, а второй обозначает позицию на текущем векторе. После файл закрывается

Код программы:

```
void P3::Task1()
{
    std::ifstream i_data("C:\\Users\\Alex\\source\\repos\\SIAOD\\Data.txt");
    unsigned long long buffer = 0;
    int maxSize = 10000000, bitArrPos;
    std::vector<unsigned long long> bitArr(maxSize / 64, 0);
    while (i_data >> buffer)
    {
        bitArrPos = buffer / 64;
        bitArr[bitArrPos] |= (1ULL << buffer % 64);
    }
    std::ofstream o_data("C:\\Users\\Alex\\source\\repos\\SIAOD\\Data.txt", std::ios::trunc);
    for (int i = 0; i < maxSize / 64; i++)
    {
        for (int y = 0; y < 64; y++)
        {
            if (((1ULL << i) & bitArr[i]) != 0)
            {
                o_data << (y + 64 * i) << ' ';
            }
        }
    }
    i_data.close();
}
```

Рис. 14 Реализация кода для задания 3.A

Драйвер и результат:

```cpp
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program3.Task1();

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

Консоль отладки Microsoft Visual Studio

```
Time: 23367mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 18096) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Data.txt – Блокнот

Файл Правка Формат Вид Справка

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 3
3 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309
9 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565
5 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821
041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1
1246 1247 1248 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266
1451 1452 1453 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471
1656 1657 1658 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676
1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881
2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086
2271 2272 2273 2274 2275 2276 2277 2278 2279 2280 2281 2282 2283 2284 2285 2286 2287 2288 2289 2290 2291
2476 2477 2478 2479 2480 2481 2482 2483 2484 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496
2681 2682 2683 2684 2685 2686 2687 2688 2689 2690 2691 2692 2693 2694 2695 2696 2697 2698 2699 2700 2701
2886 2887 2888 2889 2890 2891 2892 2893 2894 2895 2896 2897 2898 2899 2900 2901 2902 2903 2904 2905 2906
3091 3092 3093 3094 3095 3096 3097 3098 3099 3100 3101 3102 3103 3104 3105 3106 3107 3108 3109 3110 3111
3296 3297 3298 3299 3300 3301 3302 3303 3304 3305 3306 3307 3308 3309 3310 3311 3312 3313 3314 3315 3316
3501 3502 3503 3504 3505 3506 3507 3508 3509 3510 3511 3512 3513 3514 3515 3516 3517 3518 3519 3520 3521
3706 3707 3708 3709 3710 3711 3712 3713 3714 3715 3716 3717 3718 3719 3720 3721 3722 3723 3724 3725 3726
3911 3912 3913 3914 3915 3916 3917 3918 3919 3920 3921 3922 3923 3924 3925 3926 3927 3928 3929 3930 3931
4116 4117 4118 4119 4120 4121 4122 4123 4124 4125 4126 4127 4128 4129 4130 4131 4132 4133 4134 4135 4136
4321 4322 4323 4324 4325 4326 4327 4328 4329 4330 4331 4332 4333 4334 4335 4336 4337 4338 4339 4340 4341
4526 4527 4528 4529 4530 4531 4532 4533 4534 4535 4536 4537 4538 4539 4540 4541 4542 4543 4544 4545 4546
4731 4732 4733 4734 4735 4736 4737 4738 4739 4740 4741 4742 4743 4744 4745 4746 4747 4748 4749 4750 4751
4936 4937 4938 4939 4940 4941 4942 4943 4944 4945 4946 4947 4948 4949 4950 4951 4952 4953 4954 4955 4956
5141 5142 5143 5144 5145 5146 5147 5148 5149 5150 5151 5152 5153 5154 5155 5156 5157 5158 5159 5160 5161
5346 5347 5348 5349 5350 5351 5352 5353 5354 5355 5356 5357 5358 5359 5360 5361 5362 5363 5364 5365 5366
5551 5552 5553 5554 5555 5556 5557 5558 5559 5560 5561 5562 5563 5564 5565 5566 5567 5568 5569 5570 5571
5756 5757 5758 5759 5760 5761 5762 5763 5764 5765 5766 5767 5768 5769 5770 5771 5772 5773 5774 5775 5776
5961 5962 5963 5964 5965 5966 5967 5968 5969 5970 5971 5972 5973 5974 5975 5976 5977 5978 5979 5980 5981
6166 6167 6168 6169 6170 6171 6172 6173 6174 6175 6176 6177 6178 6179 6180 6181 6182 6183 6184 6185 6186
6371 6372 6373 6374 6375 6376 6377 6378 6379 6380 6381 6382 6383 6384 6385 6386 6387 6388 6389 6390 6391
6576 6577 6578 6579 6580 6581 6582 6583 6584 6585 6586 6587 6588 6589 6590 6591 6592 6593 6594 6595 6596
6781 6782 6783 6784 6785 6786 6787 6788 6789 6790 6791 6792 6793 6794 6795 6796 6797 6798 6799 6800 6801
6986 6987 6988 6989 6990 6991 6992 6993 6994 6995 6996 6997 6998 6999 7000 7001 7002 7003 7004 7005 7006
7191 7192 7193 7194 7195 7196 7197 7198 7199 7200 7201 7202 7203 7204 7205 7206 7207 7208 7209 7210 7211
7396 7397 7398 7399 7400 7401 7402 7403 7404 7405 7406 7407 7408 7409 7410 7411 7412 7413 7414 7415 7416
7601 7602 7603 7604 7605 7606 7607 7608 7609 7610 7611 7612 7613 7614 7615 7616 7617 7618 7619 7620 7621
```

Рис. 15 Драйвер, вывод задания 3.A и отсортированный файл

Задание 3.Б

Определить программный объём оперативной памяти, занимаемый битовым массивом.

Описание алгоритма:

Для подсчёта объёма нужно поделить максимальное количество чисел на результат умножения 8 * 1024 * 1024, что переводит из бит в мегабайт.

Код программы:

```
void P3::Task2()
{
    int maxSize = 10000000;
    maxSize /= (8 * 1024 * 1024);
    std::cout << maxSize << '\n';
}
```

Рис. 16 Реализация кода для задания 3.Б

Драйвер и результат:

```
#include <iostream>
#include "Practice5.hpp"
#include <chrono>

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    auto time = std::chrono::high_resolution_clock::now();

    P1 Program1;
    P2 Program2;
    P3 Program3;

    Program3.Task2();

    auto end_time = std::chrono::high_resolution_clock::now();
    auto dur = std::chrono::duration_cast<std::chrono::milliseconds>(end_time - time).count();
    std::cout << "\nTime: " << dur << "mil\n";
}
```

```
Консоль отладки Microsoft Visual Studio
1

Time: 0mil

C:\Users\Alex\source\repos\SIAOD\x64\Debug\SIAOD.exe (процесс 17864) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно…
```

Рис. 17 Драйвер и вывод задания 3.Б

Вывод

Были изучены способы приёмов сортировки с битовым представлением беззнаковых целых чисел.