****

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МИРЭА – Российский технологический университет»**

# РТУ МИРЭА

Отчет по выполнению практического задания №1

**Тема: «**Сортировка числового файла с помощью битового массива**»**

Дисциплина: **«**Структуры и алгоритмы обработки данных**»**

Выполнил студент Кузнецов А. А.

группа ИКБО-01-21

**Москва 2022**

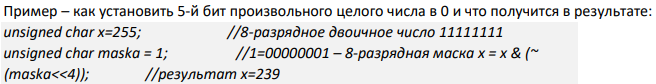
**Цель работы:**

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

**Постановка задачи**

Выполнить упражнения по применению битовых операций по изменению значений битов в ячейке оперативной памяти, созданию маски для изменения значения ячейки. Создание выражения, содержащего поразрядные операции, для выполнения определенной операции над значением ячейки.

# Задание 1.a.



Реализуйте вышеприведённый пример, проверьте правильность результата.

**Решение.** Для установки 1 в n-ый разряд исходного числа воспользуемся побитовым сдвигом маски на n разрядов влево и применим поразрядную дизъюнкцию с исходным числом.

Листинг 1 – Реализация упражнения 1.а.

int main1a() //1a

{

unsigned char x = 255; //8-разрядное двоичное число 11111111

unsigned char maska = 1; //1=00000001 – 8-разрядная маска

x = x & (~(maska << 4)); //результат x=239

cout << int(x) << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 1.



Рисунок 1 – Скриншот результата

**Задание 1.б.**

Реализуйте по аналогии с предыдущим примером решение установки 7-го бита числа в 1.

Возьмем в качестве начального числа число 128, так как в нем все биты, кроме последнего равны нулю, в том числе и седьмой.

**Решение.** Для установки 1 в n-ый разряд исходного числа воспользуемся побитовым сдвигом маски на n разрядов влево и применим поразрядную дизъюнкцию с исходным числом.

Листинг 2 – Реализация упражнения 1.б.

int main1b() //1b

{

unsigned char x = 255; //8-разрядное двоичное число 11111111

unsigned char maska = 1; //1=00000001 – 8-разрядная маска

x = x & (maska << 7); //результат x=239

cout << bitset<8>(x) << " " << int(x);

system("pause");

return 0;

}

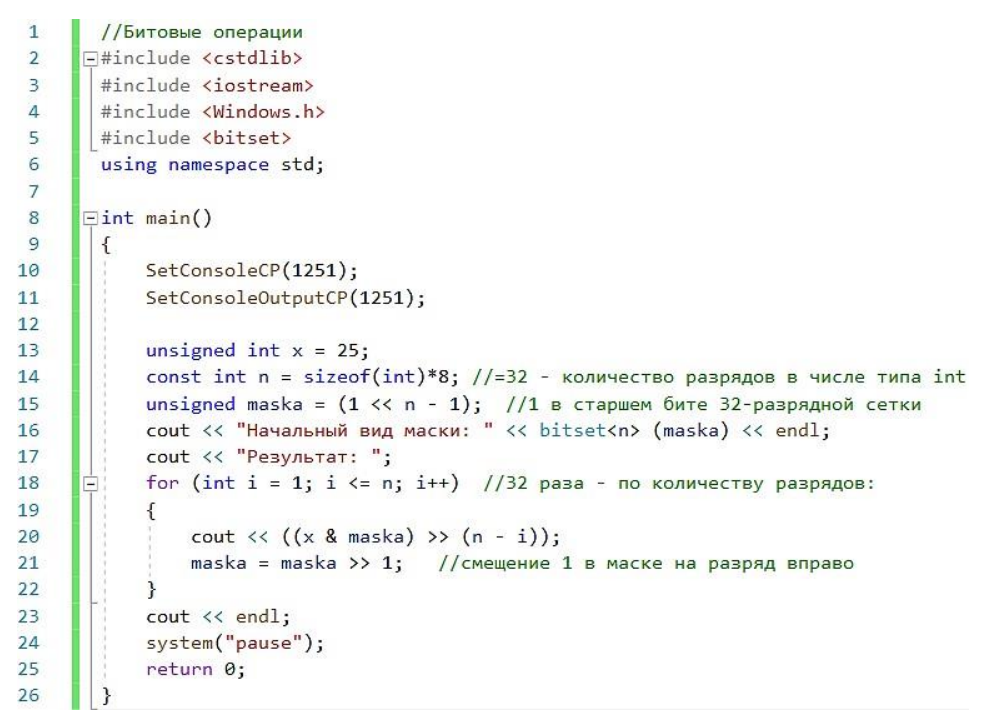
Результат выполнения программы приведен на рисунке 2.



Рисунок 2 – Скриншот результата

**Задание 1.в.**

Реализуйте код листинга, объясните полученный результат.



Листинг 3 – Реализация упражнения 1.в.

int main1v() //1v

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

unsigned int x = 25;

const int n = sizeof(int) \* 8; //=32 - количество разрядов в числе типа int

unsigned maska = (1 << n - 1); //1 в старшем бите 32-разрядной сетки

cout << "Начальный вид маски: " << bitset<n>(maska) << endl;

cout << "Результат: ";

for (int i = 1; i <= n; i++) //32 раза - по количеству разрядов:

{

cout << ((x & maska) >> (n - i));

maska = maska >> 1; //смещение 1 в маске на разряд вправо

}

cout << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 3.

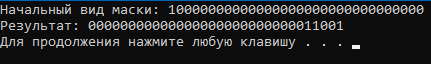


Рисунок 3 – скриншот результата

**Задание 2.а.**

Реализуйте приведённый пример с вводом произвольного набора 8-ми чисел (со значениями от 0 до 7) и его сортировкой с помощью битового массива. Проверьте его работу. Если количество чисел в исходной последовательности больше 8 и/или значения превосходят 7, можно подобрать тип беззнакового числа для битового массива с подходящим размером разрядной сетки (до 64 в типе unsigned long long).

Листинг 4 – Реализация упражнения 2.а.

int main2a() //2a

{

unsigned char arr = 0;

unsigned char maska = 1;

unsigned char randArr[6] = { 1, 0, 5, 7, 2, 4 };

for (int i = 0; i < 6; i++)

arr = arr | (maska << (7 - randArr[i]));

cout << bitset<8>(arr) << endl;

int j = 5;

for (int i = 0; i < 8; i++)

{

if (bitset<8>(arr)[i] == 1)

{

randArr[j] = (7 - i);

j--;

if (j == -1)

break;

}

}

for (int i = 0; i < 6; i++)

cout << int (randArr[i]) << " ";

cout << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 4.



Рисунок 4 – скриншот результата

**Задание 2.б.**

Адаптируйте вышеприведённый пример для набора из 64-х чисел (со значениями от 0 до 63).

Листинг 5 – Реализация упражнения 5

int main2b() //2b

{

unsigned long long arr = 0;

unsigned long long maska = 1;

int randArr[7] = { 10, 7, 56, 43, 23, 49, 63 };

for (int i = 0; i < 7; i++)

arr = arr | (maska << (63 - randArr[i]));

cout << bitset<64>(arr) << endl;

int j = 6;

for (int i = 0; i < 64; i++)

{

if (bitset<64>(arr)[i] == 1)

{

randArr[j] = (63 - i);

j--;

if (j == -1)

break;

}

}

for (int i = 0; i < 7; i++)

cout << int(randArr[i]) << " ";

cout << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 5.



Рисунок 5 – скриншот результата

**Задание 2.в.**

Исправьте программу задания 2.б, чтобы использовалось не одно число типа unsigned long long, а линейный массив чисел типа unsigned char.

Листинг 5 – Реализация упражнения 5

int main2v() //2v

{

bitset<256> arr = 0;

bitset<256> maska = 1;

unsigned char randArr[8] = { 105, 57, 255, 56, 0, 55, 3, 216 };

for (int i = 0; i < 8; i++)

arr = arr | (maska << (255 - randArr[i]));

cout << arr << endl;

int j = 7;

for (int i = 0; i < 256; i++)

{

if (bitset<256>(arr)[i] == 1)

{

randArr[j] = (255 - i);

j--;

if (j == -1)

break;

}

}

for (int i = 0; i < 8; i++)

cout << int(randArr[i]) << " ";

cout << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 5.

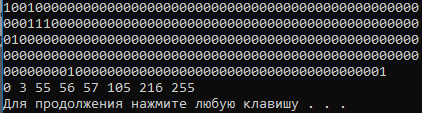


Рисунок 5 – скриншот результата

**Задание 3[a-b]**

Будем использовать 1 мб оперативной памяти битового массива (размер стека). Наша маска будет равна 1. А далее воспользуемся методами set и test битового массива (bitset) для установки и проверки разрядов битового массива. Код программы представлен в листинге 5.

Листинг 5 – Реализация упражнения 5

int main3() {

time\_t start;

time\_t end;

const int n = 8 \* 1024 \* 1024; // 1 mb = 8 388 608 bit

bitset<n>\* arr = new bitset<n>(0);

int num;

int sum = 0;

ifstream file("random.txt");

while (file.is\_open()) {

time(&start);

while (file >> num) {

if (num < n) // (num < 8 388 608)

arr->set(num);

sum++;

}

time(&end);

file.close();

}

cout << "Sort time: " << end - start << endl;

ofstream nwFile("file.txt");

if (nwFile.is\_open()) {

for (int i = 0; i < n; i++) {

if (arr->test(i)) {

nwFile << i << " ";

}

}

nwFile.close();

}

cout << "Number of numbers read: " << sum << endl;

cout << "Array size (buts): " << arr->size() << endl;

system("pause");

return 0;

}

Результат выполнения программы приведен на рисунке 5.

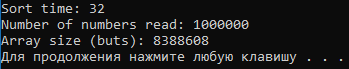


Рисунок 5 – скриншот результата

Как мы видим, для 1000000 значений чисел, соответствующих условию, потребовалось 32 секунды на сортировку. Пример вида файла приведен на рисунке 6.

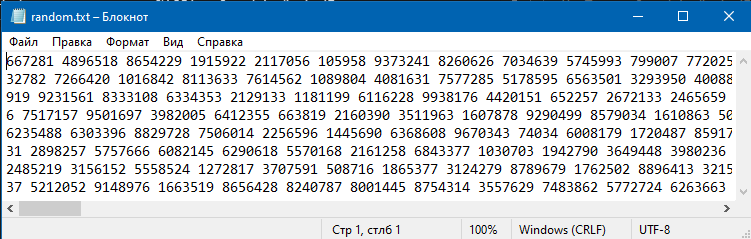


Рисунок 6 – файл рандомных чисел

**Вывод**

В ходе работы были выполнены поставленные задачи и получены навыки программирования на языке C++. В результате выполнения практической работы были получены знания по реализации побитовых операций на примере данных упражнений.

**Список информационных источников**

1. Лекции по структурам и алгоритмам обработки данных
2. Методическое пособие Архитектура ВМиС часть 1.