|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3** | |
| **по дисциплине** | |
| **«**Структуры и алгоритмы обработки данных**»**  **Тема: «Поразрядные операции и их применение»** | |
|  | |
| Выполнил студент группы ИКБО-03-21 | Хречко С.В. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2022

# **Цель работы**

Получение навыков применения поразрядных операций в алгоритмах.

# **Постановка задачи**

1. Разработать программу, которая продемонстрирует выполнение упражнений варианта. Результаты выполнения упражнения выводить на монитор.

Требования к упражнениям:

1. Определить переменную целого типа, присвоить ей значение, используя константу в шестнадцатеричной системе счисления. Разработать функцию, которое установит заданные в задании биты исходного значения переменной в значение 1, используя соответствующую маску и поразрядную операцию.
2. Разработать функцию, которая обнуляет заданные в задании биты исходного значения целочисленной переменной, введенной пользователем, используя соответствующую маску и поразрядную операцию.
3. Разработать функцию, которая умножает значение целочисленной переменной, введенной пользователем, на множитель, используя соответствующую поразрядную операцию.
4. Разработать функцию, которая делит значение целочисленной переменной, введенной пользователем, на делитель, используя соответствующую поразрядную операцию.
5. Разработать функцию, реализующую задание, в которой используются только поразрядные операции. В выражении используется маска – переменная. Маска инициализируется единицей в младшем разряде (маска 1) или единицей в старшем разряде (маска 2). Изменяемое число и n вводится с клавиатуры.
6. Провести тестирование программы на небольших объемах данных, введенных вручную. Разработанные тесты должны покрывать все случаи входных данных (средний, лучший, худший). Результаты тестирования свести в сводные таблицы.
7. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Вариант №1. Условие задания:

|  |  |
| --- | --- |
| Упражнение 1 | 5-ый и 7-ой справа |
| Упражнение 2 | С 9-ого четыре слева |
| Упражнение 3 | 8 |
| Упражнение 4 | 8 |
| Упражнение 5 | Установить n-ый бит в 1, используя маску 1 |

# **Решение**

\*немного теории по теме работы в целом\*

\*немного теории про поразрядные операции\*

\*объяснение алгоритма написанных функций + полученный код функций (основной фрагмент, не вся программа)\*

|  |
| --- |
| //Функция умножения с помощью поразрядных операций (упр. 3)  void bitmult()  {  } |

\*описание работы программы (пользовательского интерфейса) + скриншот интерфейса\*

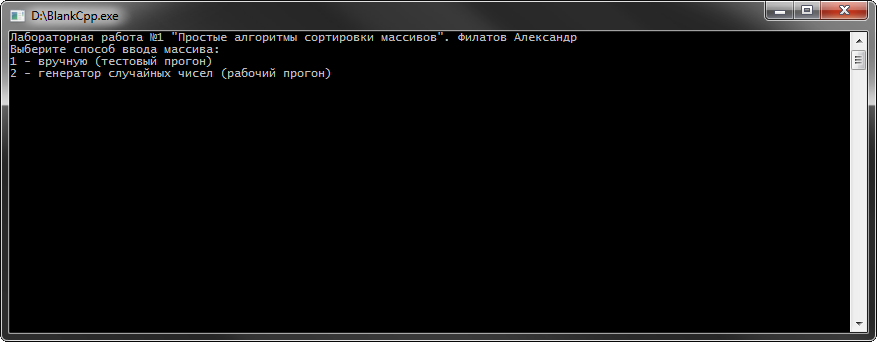


Рисунок 1. Интерфейс программы

# **Тестирование**

\*описание **тестового прогона** программы + скриншот. Тестовый прогон – прогон на маленьком объеме входных данных, введенных вручную, программа должна выводить на экран ход решения, чтобы можно было это проверить\*

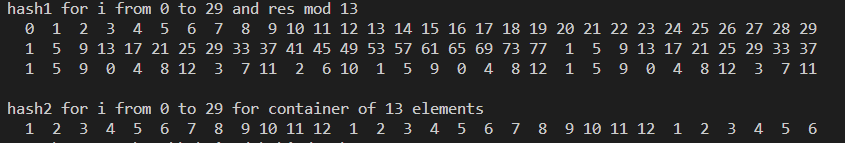


Рисунок 1. Тестирование Хэш-функций

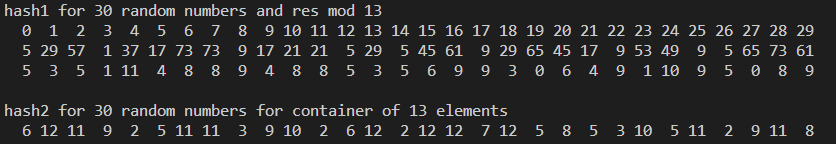


Рисунок 1. Тестирование Хэш-функций

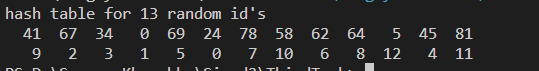


Рисунок 1. Тестирование функции добавления

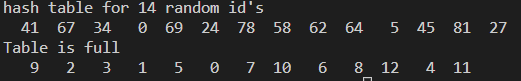


Рисунок 1. Тестирование функции добавления

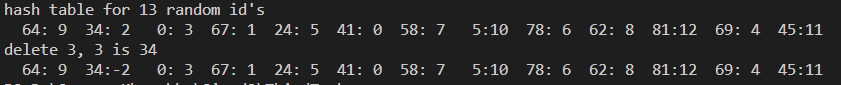


Рисунок 1. Тестирование функции удаления

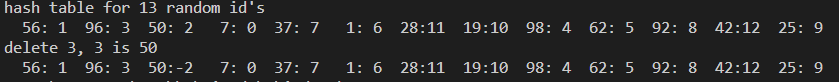


Рисунок 1. Тестирование функции удаления

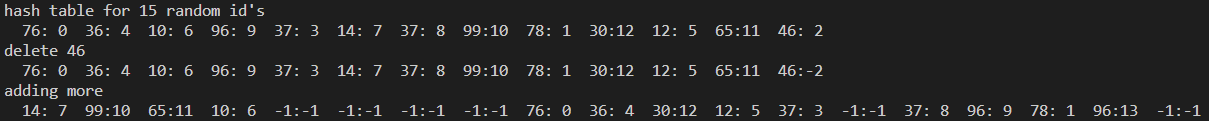


Рисунок 1. Тестирование добавления с рехэшированием

На след. Рис. Видно, что после удаления 40-ка, на позиции 1 в таблице (удаление проверено, присваивает -2), поиск пошел искать дальше, и добавив 3 (hash2 для 20), нашел искомое.

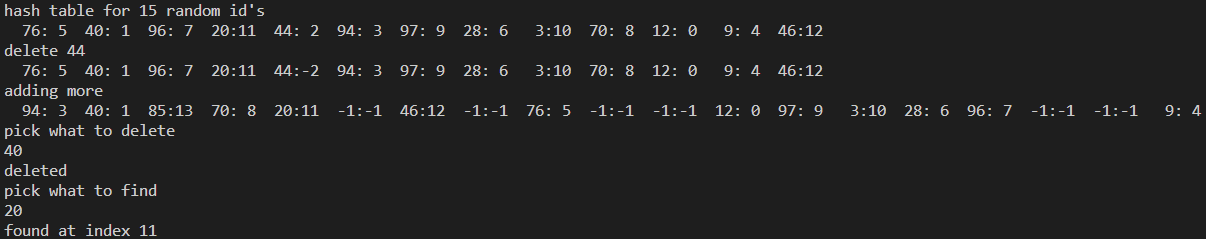


Рисунок 1. Тестирование поиска

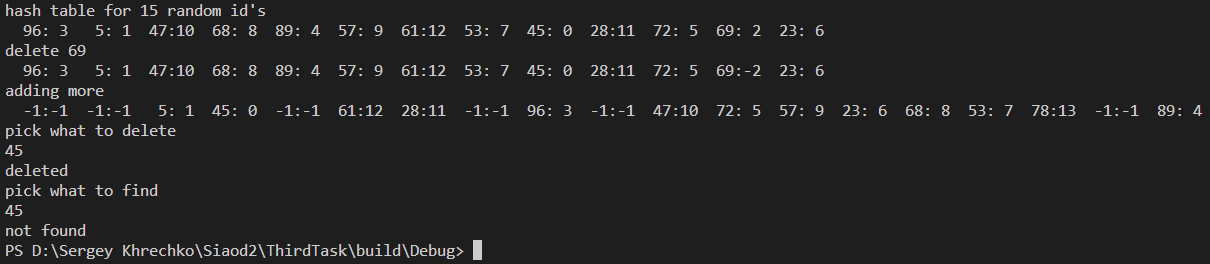


Рисунок 1. Тестирование поиска

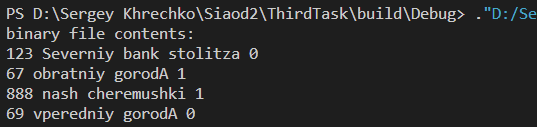


Рисунок 1. Тестирование перевода в бинарный файл

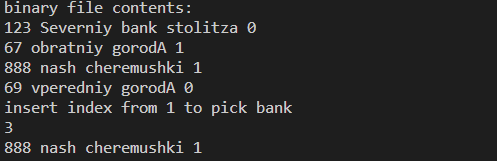


Рисунок 1. Тестирование получения записи файла

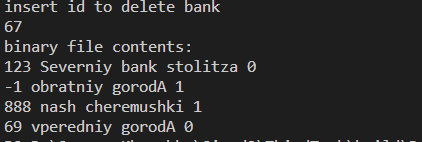


Рисунок 1. Тестирование удаления записи по id

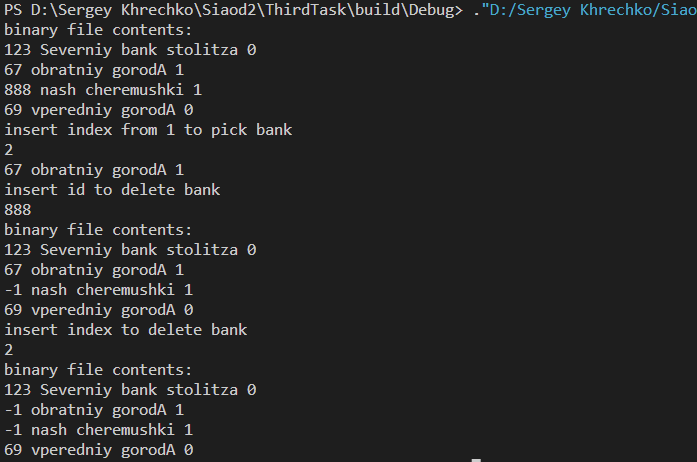


Рисунок 1. Тестирование удаления записи по индексу

Мы добавляем на позицию 2 и в конец (на позиции два уже есть запись => заменяем, это для записи в удаленные позиции, если вдруг понадобится)

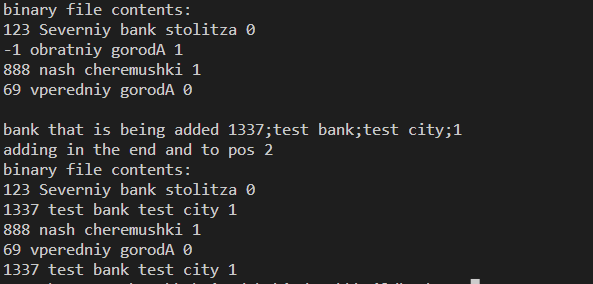


Рисунок 1. Тестирование добавления записи

Из результатов выполнения программы видно:

1. \*выводы, которые можно сделать на основе полученных результатов\*

# **Вывод**

В результате выполнения работы я:

1. Освоил алгоритмы работы с поразрядными операциями и их реализацию на языке программирования C++
2. Научился программировать автоматическое тестирование простых программ

# **Исходный код программы**

|  |
| --- |
| void bitmult()  {  }  int main()  {  } |