

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

КУРСОВАЯ РАБОТА
по дисциплине «Алгоритмы и структуры данных»
Тема: Хеш-таблицы с открытой адресацией

Студентка гр. 8381

Преподаватель

Ивлева О.А.

Жангиров Т.Р.

Санкт-Петербург

2019

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студентка Ивлева О.А.

Группа 8381

Тема работы: Хеш-таблицы с открытой адресацией

Исходные данные: необходимо реализовать демонстрацию работы алгоритма вставки и удаления в хеш-таблице с открытой адресацией, включающую пошаговое выполнение алгоритма или сразу до конца по выбору пользователя. Пошаговое выполнение включает шаг вперед и шаг назад с текстовым выводом и наглядным рисованием хеш-таблицы.

Содержание пояснительной записки:

«Содержание», «Введение», «Задание», «Описание программы», «Тестирование», «Демонстрация», «Заключение», «Список использованных источников».

Предполагаемый объем пояснительной записки:

Не менее 60 страниц.

Дата выдачи задания:

Дата сдачи реферата:

Дата защиты реферата:

Студентка

Ивлева О.А.

Преподаватель

Жангиров Т.Р.

АННОТАЦИЯ

В ходе выполнения курсовой работы была разработана программа с GUI, позволяющая наглядно показывать алгоритм вставки и удаления в хеш-таблице с открытой адресацией. Программа обладает следующей функциональностью: создание хеш-таблицы по заданному файлу, создание хеш-таблицы с данными, введенными пользователем, хеш-функция и динамическое расширение размера хеш-таблицы; демонстрация работы вставки и удаления алгоритма.

SUMMARY

In the course work, a program was developed with a GUI that allows you to clearly show the insert and delete algorithm in a hash table with open addressing. The program has the following functionality: creating a hash table for a given file, creating a hash table with data entered by the user, a hash function and dynamically expanding the size of the hash table; demonstration of the insertion and deletion of the algorithm.

СОДЕРЖАНИЕ

	Введение	6
1.	Задание	7
2.	Описание программы	8
2.1.	Описание интерфейса пользователя	8
2.2.	Описание основных классов для хеш-таблицы	10
2.3.	Описание алгоритма вставки в хеш-таблицу	12
2.4.	Описание класса StrStrWorker для работы с хеш-таблицей	12
2.5.	Описание структур и функций для исследования	14
3	Тестирование	16
3.1.	Вид программы	16
3.2.	Тестирование генерации файла	17
3.3.	Тестирование создания хеш-таблицы	18
4	Исследование	21
4.1	План экспериментального исследования	21
4.2	Технология проведения исследования	22
4.3	Исследование зависимостей от максимальной длины ключа	24
4.4	Исследование зависимостей от множителя размера таблицы	26
4.5	Исследование зависимостей от числа пар	29
4.6	Вывод об эффективности хеш-функций	34
4.7	Исследование худшего случая вставки	35
4.8	Выводы об исследовании алгоритма	37
	Заключение	38
	Список использованных источников	39
	Приложение А. Исходный код программы. main.c	40
	Приложение Б. Исходный код программы. pair.h	41
	Приложение В. Исходный код программы. cvector.h	46
	Приложение Г. Исходный код программы. hashtable.h	48

Приложение Д. Исходный код программы. strstrworker.h	53
Приложение Е. Исходный код программы. strstrworker.cpp	54
Приложение Ж. Исходный код программы. strstrtester.h	60
Приложение И. Исходный код программы. strstrtester.cpp	61
Приложение К. Исходный код программы. mainwindow.h	65
Приложение Л. Исходный код программы. mainwindow.cpp	66
Приложение М. Исходный код программы. libraries.h	70
Приложение Н. Исходный код программы. lr5.pro	71
Приложение О. Исходный код программы. mainwindow.ui	72

ВВЕДЕНИЕ

Цель работы

Реализация и демонстрация алгоритмов кодирования (Фано-Шеннона, Хаффмана), быстрого поиска на основе БДП или хеш-таблиц, сортировок.

Основные задачи

Визуализация структур данных, алгоритмов, действий. Демонстрация должна быть подробной и понятной (в том числе сопровождаться пояснениями), чтобы программу можно было использовать в обучении для объяснения используемой структуры данных и выполняемых с ней действий.

Методы решения

Разработка программы велась на базе операционной системы Windows 10 в среде разработки QtCreator. Для создания графической оболочки использовался редактор интерфейса в QtCreator и система сигналов-слотов Qt. Для реализации работы с хеш-таблицей был создан класс Hash_cl, динамический массив структур val. Для работы таблицы с ключами и значениями в формате строк был создан файл functions. Для графического изображения хеш-таблицы использовалась библиотека QGraphicsScene.

1. ЗАДАНИЕ

Необходимо провести визуализацию алгоритма вставки и исключения в хеш-таблице с открытой адресацией.

Демонстрация включает:

- Визуализацию структур данных, алгоритмов, действий.
- подробный и понятный (в том числе сопровождаться пояснениями) вывод работы алгоритма, чтобы программу можно было использовать в обучении для объяснения используемой структуры данных и выполняемых с ней действий.

2. ОПИСАНИЕ ПРОГРАММЫ

2.1. Описание интерфейса пользователя

Интерфейс программы состоит из поля ввода новых элементов для вставки или исключения; кнопок для моментального вывода или пошагового выполнения; кнопки для полного очищения хеш-таблицы. Основные виджеты панели генерации файла и их назначение представлены в табл. 1.

Таблица 1 – Основные виджеты интерфейса

Класс объекта	Название виджета	Назначение
QLineEdit	Input	Поле ввода элементов
QPushButton	Push_File	Вставка элементов из файла
QPushButton	Del_all	Очистить хеш-таблицу
QPushButton	NextStep	Шаг вперед для вставки
QPushButton	Next_del	Шаг вперед для удаления
QPushButton	Del_elem	Исключение выбранных элементов
QPushButton	PrevStep	Шаг назад для вставки
QPushButton	Prev_del	Шаг назад для удаления
QLabel	Output	Текстовая демонстрация выполнения
QGraphicsView	GraphicsView	Рисование хеш-таблицы

2.2. Описание основных классов для хеш-таблицы

Для реализации хеш-таблицы был создан шаблонный класс Hash_cl. Основные методы класса представлены в табл. 2.

Таблица 2 – Основные методы класса Hash_cl

Метод	Назначение
int get_size()	Возвращает размер хеш-таблицы
int get_last_added()	Возвращает индекс последнего добавленного элемента
int get_last_repeat()	Возвращает индекс последнего повторяющегося элемента
void reset_last_repeat()	Возвращает флаг, что элемент больше не повторяется
void delete_all()	Очищает хеш-таблицу
void resize(int sum)	Если хеш-функция больше размера на данный момент, то происходит увеличение массива
void print_hash(string &output)	Текстовая запись хеш-таблицы на данный момент

Основные методы класса Hash_cl:

```

struct Hash {
    Key key1;
    Value value1;
    int h_func = 0;
    int flag_repeat = 0;
    int fl_del = 0;
};

```

Key – отвечает за ключ пары

Value – отвечает за значение пары

H_func – значение хеш-функции

Flag_repeat – флаг повтора элемента для алгоритма вставки

Fl_del – флаг удаления элемента, для алгоритма исключения

int size = 100 – объявление начального размера массива

Hash *val = new Hash[size] – динамический массив элементов

int last_added = -1 – индекс последнего добавленного элемента

int last_repeat = -1 – индекс последнего повторяющегося элемента

Метод do_hash_str(Key key, Value value, int h_sum, string &output), получает на вход ключ, значение, хеш-функцию и адрес строки вывода. Если значение хеш-функции превышает размер, то вызывается метод resize, которому передается значение хеш-функции + 100. Далее происходит сравнение, если ячейка массива под номером значения хеш-функции свободна, то происходит запись в эту ячейку и записывается, что это последний добавленный элемент, если же данная ячейка занята, то мы проходим дальше по массиву, пока не найдем свободную ячейку. Если встретим по пути элемент с таким же ключом, то поиск свободной ячейки завершается и записывается в строку сообщение о том, что такой ключ уже есть.

Метод do_hash_del(Key key, Value value, int h_sum) получает на вход ключ, значение и хеш-функцию. Далее происходит присваивание индексу i хеш-функции и происходит поиск по массиву данного элемента. Если поиск прошел успешно, то элемент удаляется, значения возвращаются в исходное положение.

2.3. Описание алгоритма вставки в хеш-таблицу

Хеш-таблица содержит массив, элементы которого есть пары. Выполнение вставки в хеш-таблицу начинается с вычисления хеш-функции. Алгоритм вставки элемента последовательно проверяет ячейки массива, начиная с ячейки с индексом значения хеш-функции, пока не будет найдена первая свободная ячейка, в которую и будет записан новый элемент. Используется линейное пробирование, то есть ячейки хеш-таблицы последовательно просматриваются

друг за другом, с единичным интервалом. Пример коллизии с разрешением открытой адресацией представлен на рис. 1.

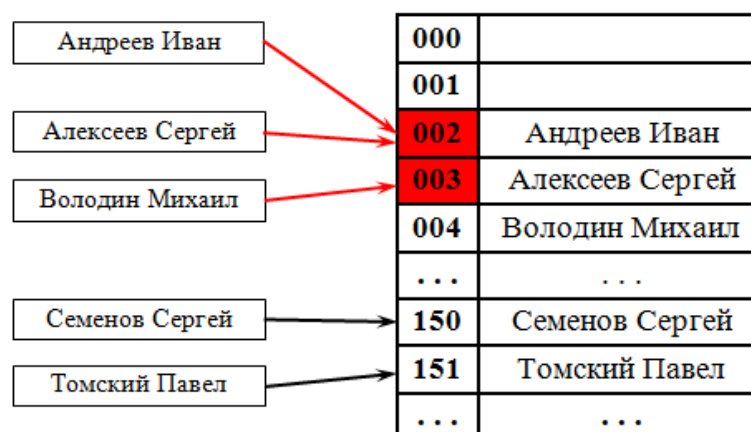


Рисунок 1 – Пример коллизии в открытой адресации

Здесь стрелки показывают значение хеш-функции для ключа. Так как у ключей «Андреев Иван» и «Алексеев Сергей» одинаковое значение хеш-функции, последний добавленный размещается на следующую ячейку. Так как ячейка массива по значению хеш-функции ключа «Володин Михаил» занята, то этот ключ также размещается на следующей ячейке.

2.3. Описание алгоритма исключения из хеш-таблицы

Хеш-таблица содержит массив, элементы которого есть пары. Выполнение исключения из хеш-таблицы начинается с проверки элементов с индекса хеш-функции данного элемента. Алгоритм исключения элемента последовательно проверяет ячейки массива, начиная с ячейки с индексом значения хеш-функции, пока не будет найдена подходящая ячейка, если такая имеется, в которой и будет удален элемент. Используется линейное пробирование, то есть ячейки хеш-таблицы последовательно просматриваются друг за другом, с единичным интервалом.

2.4. Описание функций `functions` для работы с хеш-таблицей

Для демонстрации работы алгоритмов хеширования, вставки и исключения из хеш-таблицы для ключей и значений был выбран формат строк. Методы представлены в табл. 3.

Таблица 3 – Методы для работы со строкой

Название функции	Описание
<code>void open(string &str);</code>	Добавление элементов из файла.
<code>int sum(string strNew);</code>	Вычисление хеш-функции Хеш-функция представляет из себя сумму всех символов ключа пары.

Функция `make_str(Hash_cl<string, string> &Hash_, string &str, string &output, int oper)`, которая получает на вход адрес класса, адрес считанной строки, адрес выходной строки и флаг, который определяет запущена ли эта функция из вставки или удаления. Сначала объявляются строки для ключа и значения, флаг, и переменная для значения хеш-функции. Далее идет проход по строке, если встречен символ `|` - разделитель элементов, то это означает, что нужно вычислять значение хеш-функции и создавать данную пару в классе. Если встречен символ `:` - разделитель ключ-значение, то это показывает, что следующие символы нужно считывать в значение, для этого значение флага становится 1.

Функция `make_str_step(Hash_cl<string, string> &Hash_st, string &str_step, string &output, int &ind_st, int oper)`, которая получает на вход адрес класса, адрес считанной строки для вывода по шагам, адрес выходной строки и флаг, который определяет запущена ли эта функция из вставки или удаления. Сначала объявляются строки для ключа и значения, флаг, и переменная для хранения значения хеш-функции. Далее идет проход по строке до первого встреченного символа `|` или `\0` для демонстрации выполнения по шагам. Если встречен символ `:`

- разделитель ключ-значение, то флаг становится равным 1 и следующие символы считываются в значение. Далее происходит вычисление хеш-функции и определение, какой метод в классе должен вызываться, для создания или исключения элемента, в зависимости от операции, выбранной пользователем.

Функция `make_str_del(Hash_cl<string, string> &Hash_st, string &str_step, string &output, int &ind_st, int oper)`, которая получает на вход адрес класса, адрес считанной строки для вывода по шагам, адрес выходной строки и флаг, который определяет запущена ли эта функция из вставки или удаления. Сначала объявляются строки для ключа и значения, флаг, и переменная для хранения значения хеш-функции. Далее идет проход по строке до первого встреченного символа `|` или `\0` для демонстрации выполнения по шагам. Если встречен символ :
- разделитель ключ-значение, то флаг становится равным 1 и следующие символы считываются в значение. Далее происходит вычисление хеш-функции и определение, какой метод в классе должен вызываться, для создания или исключения элемента, в зависимости от операции, выбранной пользователем. Отличие в том, что передвижение по строке происходит не вправо, а влево, то есть так мы понимаем, какой элемент нам нужно удалить или вернуть.

2.5 Описание рисования хеш-таблицы

Функция `QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene *&scene)` обнуляет сцену, задает цвет для рисования прямоугольников, шрифт добавляемых значений и вызывает саму функцию рисования.

Выбранный стиль окончаний `Qt::RoundCap`:



Выбранный стиль соединений Qt::RoundJoin:



Функция `add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1, int y1, int h, int w, QPen &pen, QBrush &brush, QFont &font)`, отвечает за рисование хеш-таблицы. Индекс i проходит по всему массиву ключ-значение, и если элемент не пуст, то рисуется прямоугольник зеленого цвета, если такой ключ уже существует, то прямоугольник этого ключа обводится красным цветом. Далее в середину прямоугольника вставляется текст с ключом и значением добавленного элемента. Если происходит исключение элемента, то прямоугольник удаляется. Для добавления нового прямоугольника значение y увеличивается на 40 пикселей.

3. ТЕСТИРОВАНИЕ

3.1. Вид программы

Программа представляет собой окно с графическим интерфейсом шириной 1098 пикселей и высотой 800 пикселей. Вид программы после запуска представлен на рис. 2.

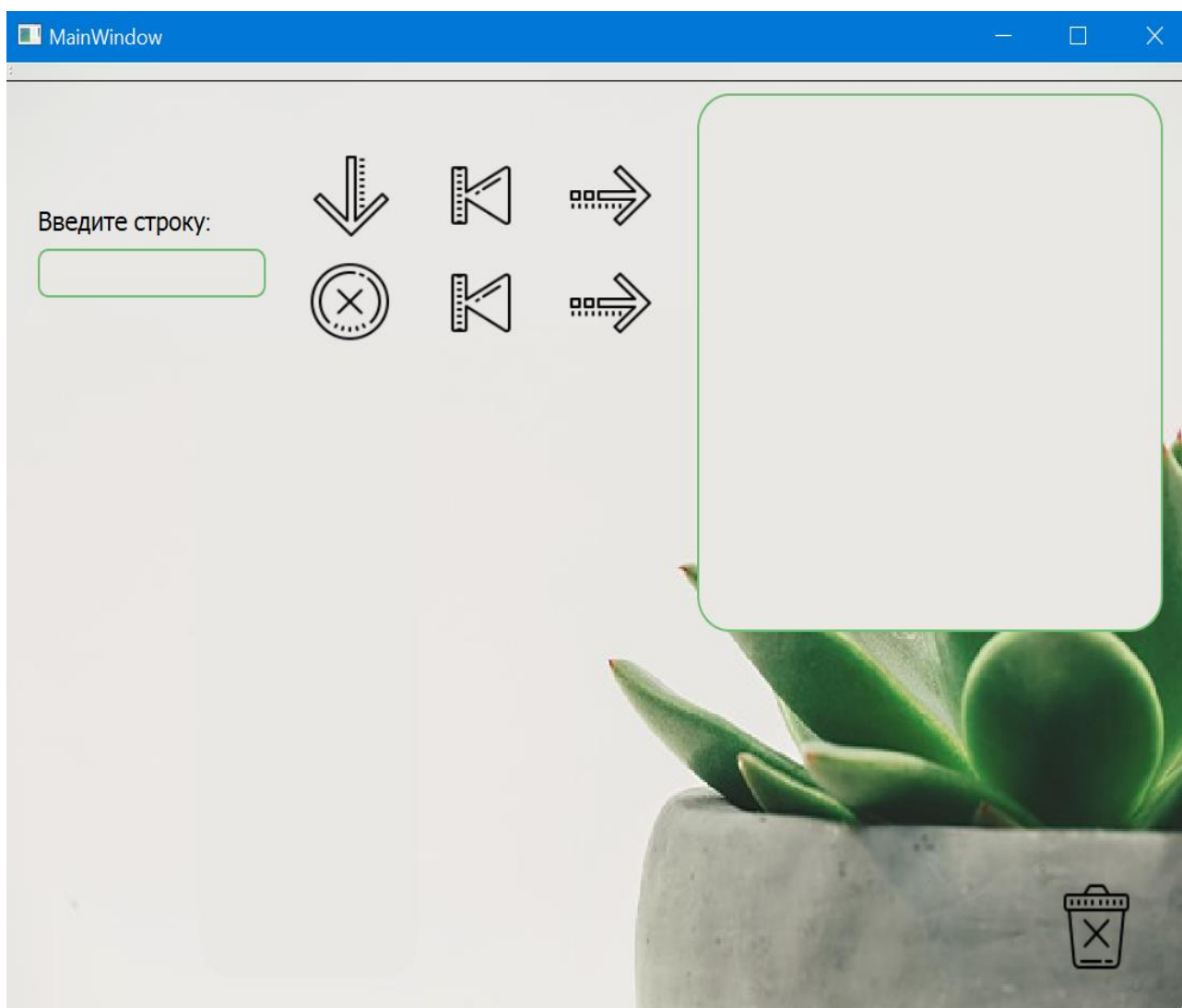


Рисунок 2 – Вид программы после запуска

Далее при нажатии стрелки вниз, элементы из файла загружаются в хеш-таблицу.

Вид программы после создания хеш-таблицы представлен на рис. 3.

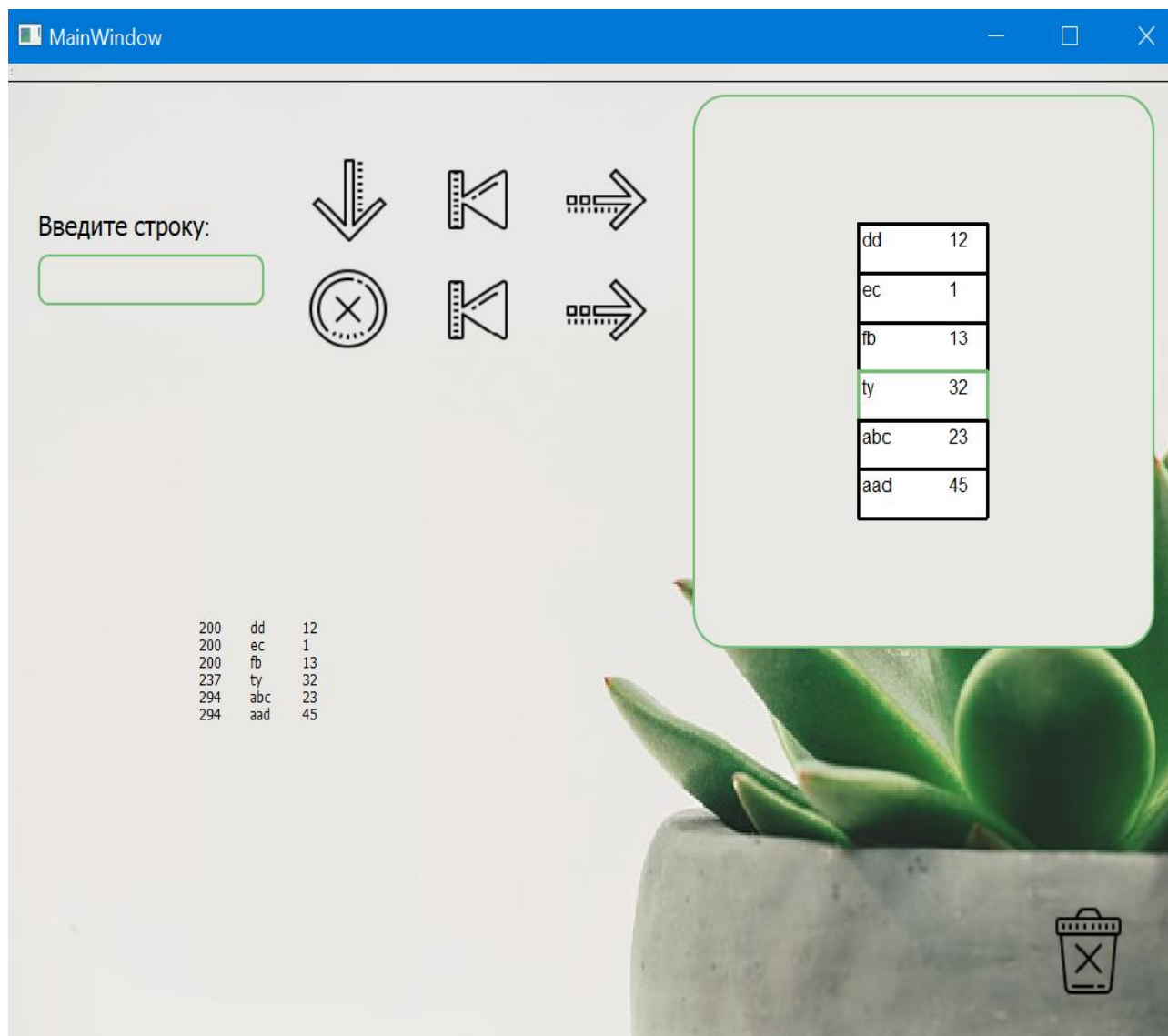


Рисунок 3 – Вид программы после создания хеш-таблицы

Вид программы с добавлением элементов представлен на рис. 4.

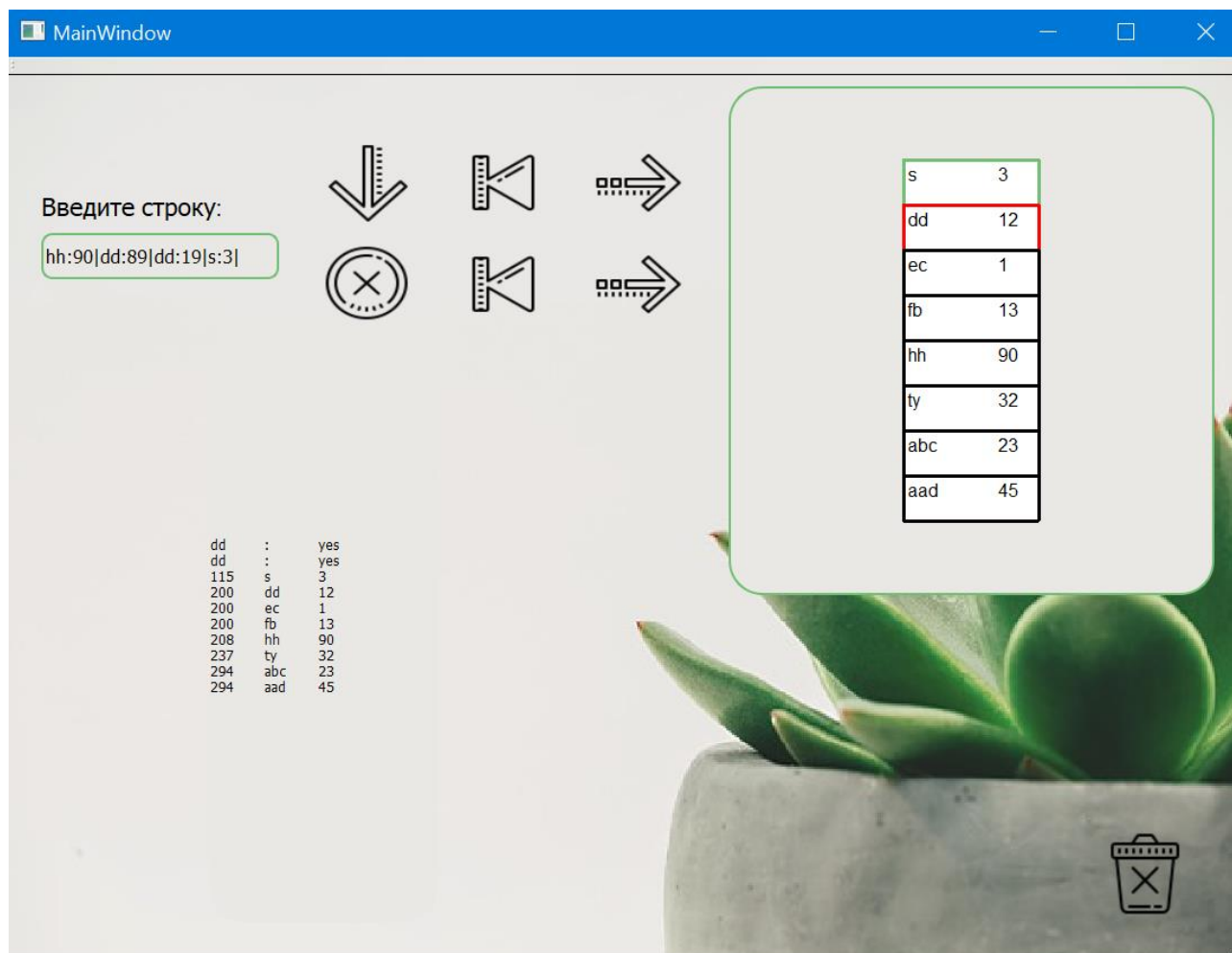


Рисунок 4 – Вид программы с добавлением элементов

Вид программы с добавлением элементов представлен на рис. 5.

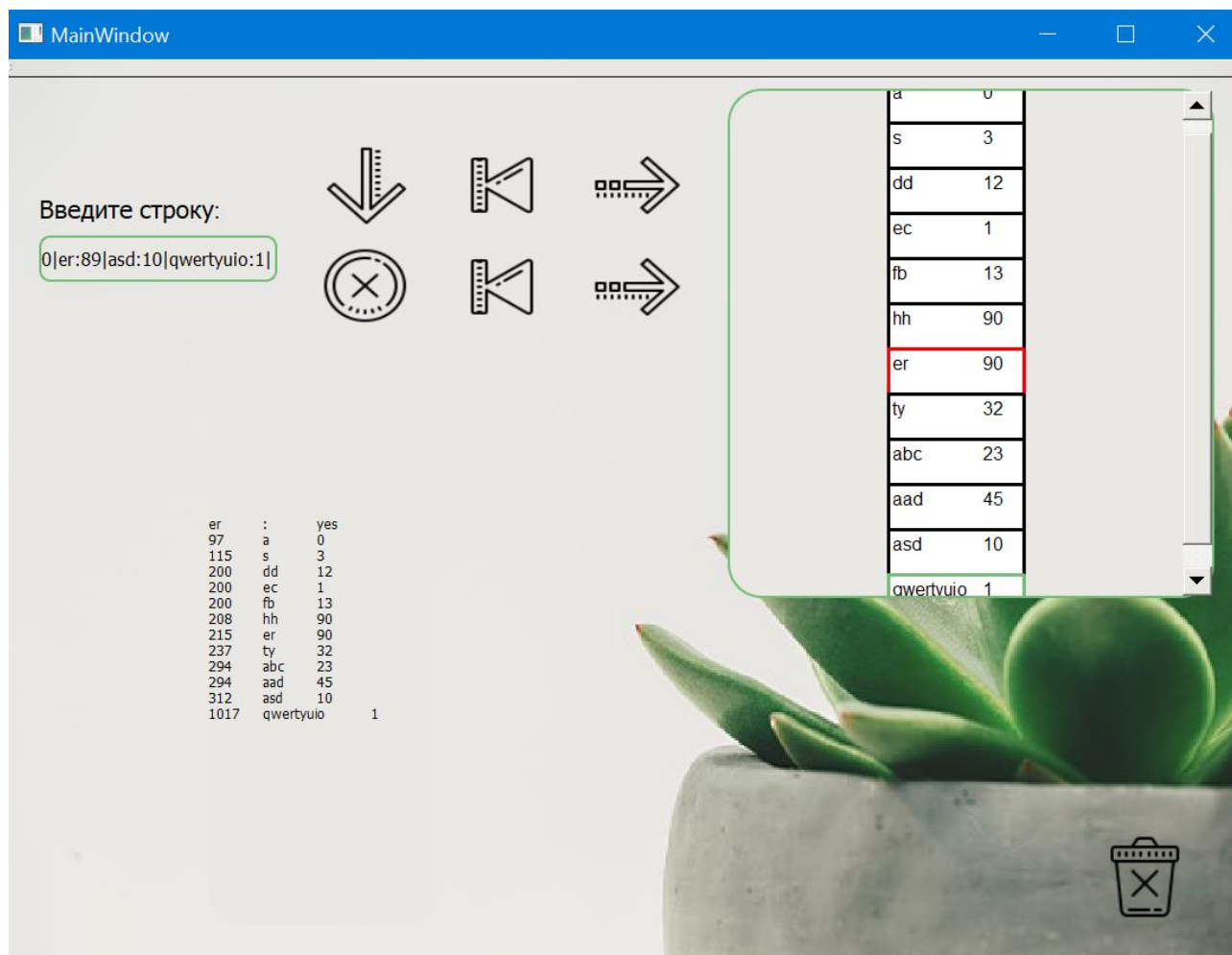


Рисунок 5 – Вид программы с добавлением элементов

Вид программы после очищения хеш-таблицы представлен на рис. 6.

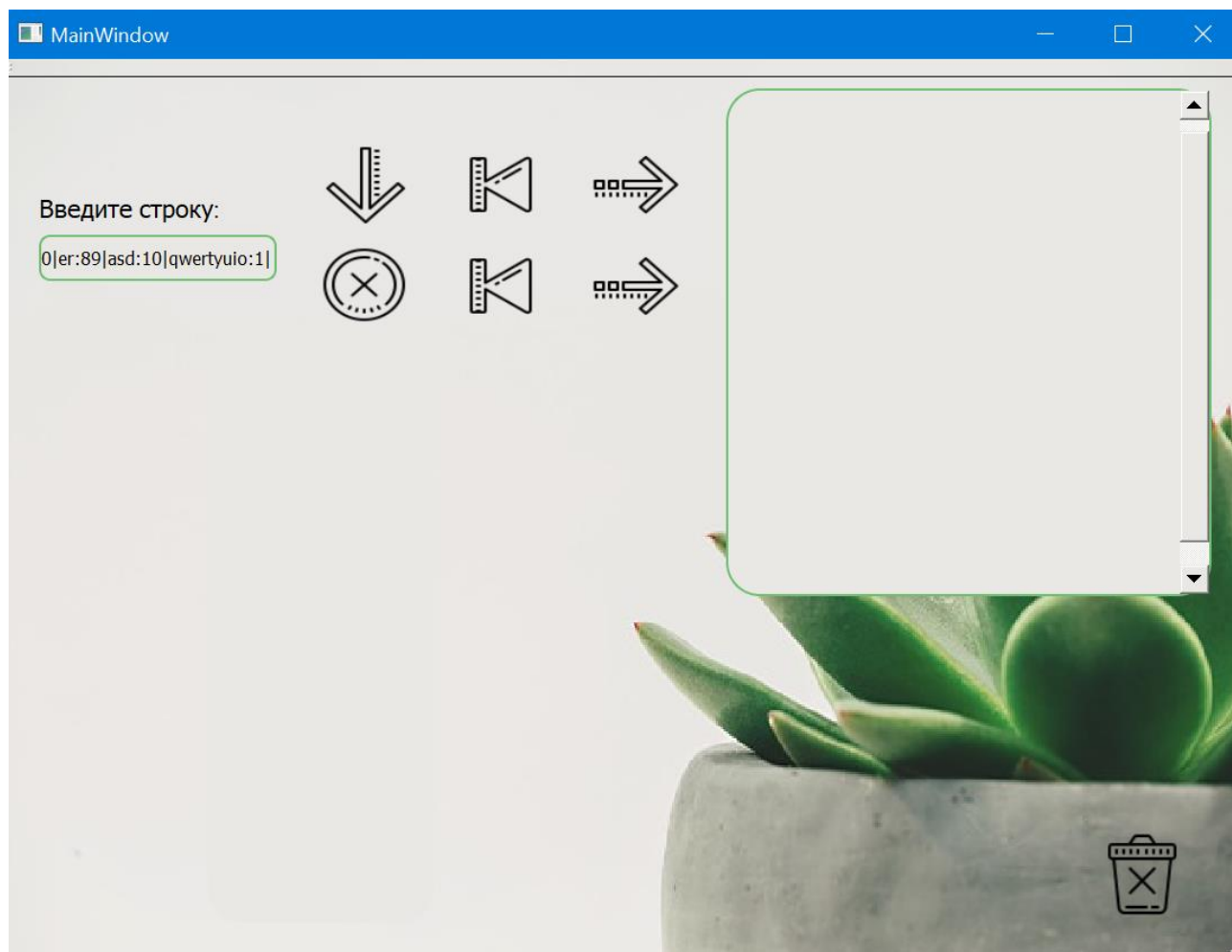
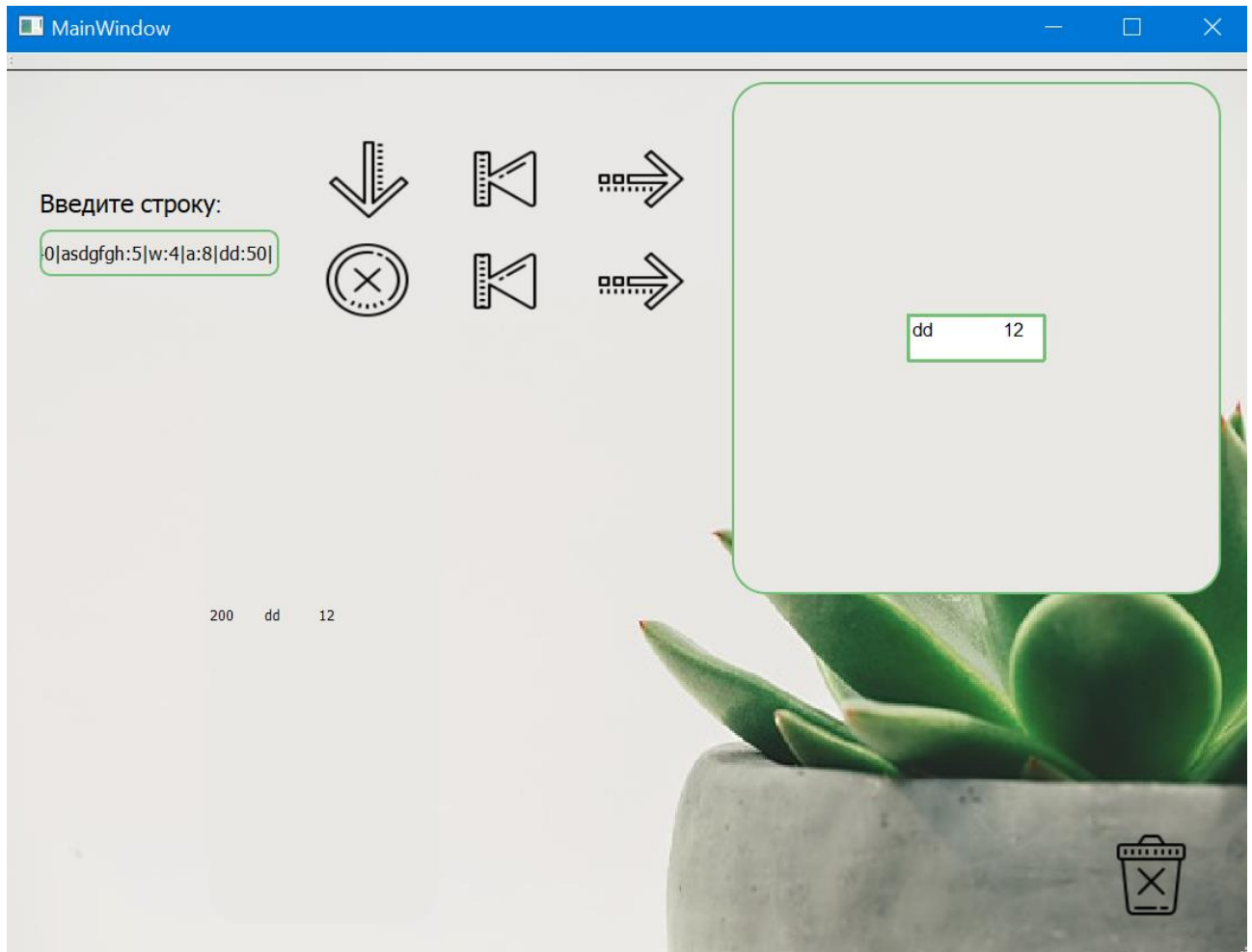
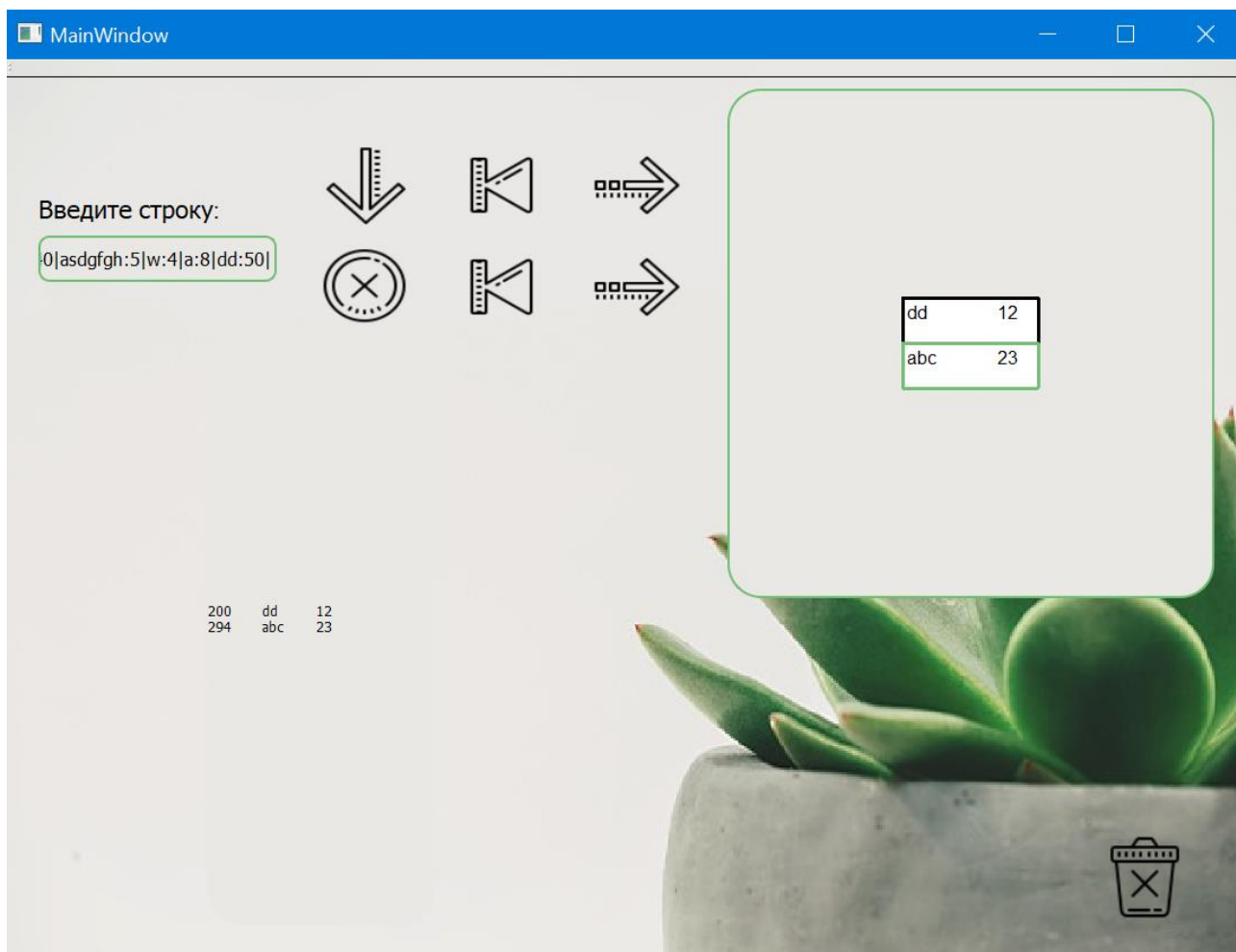


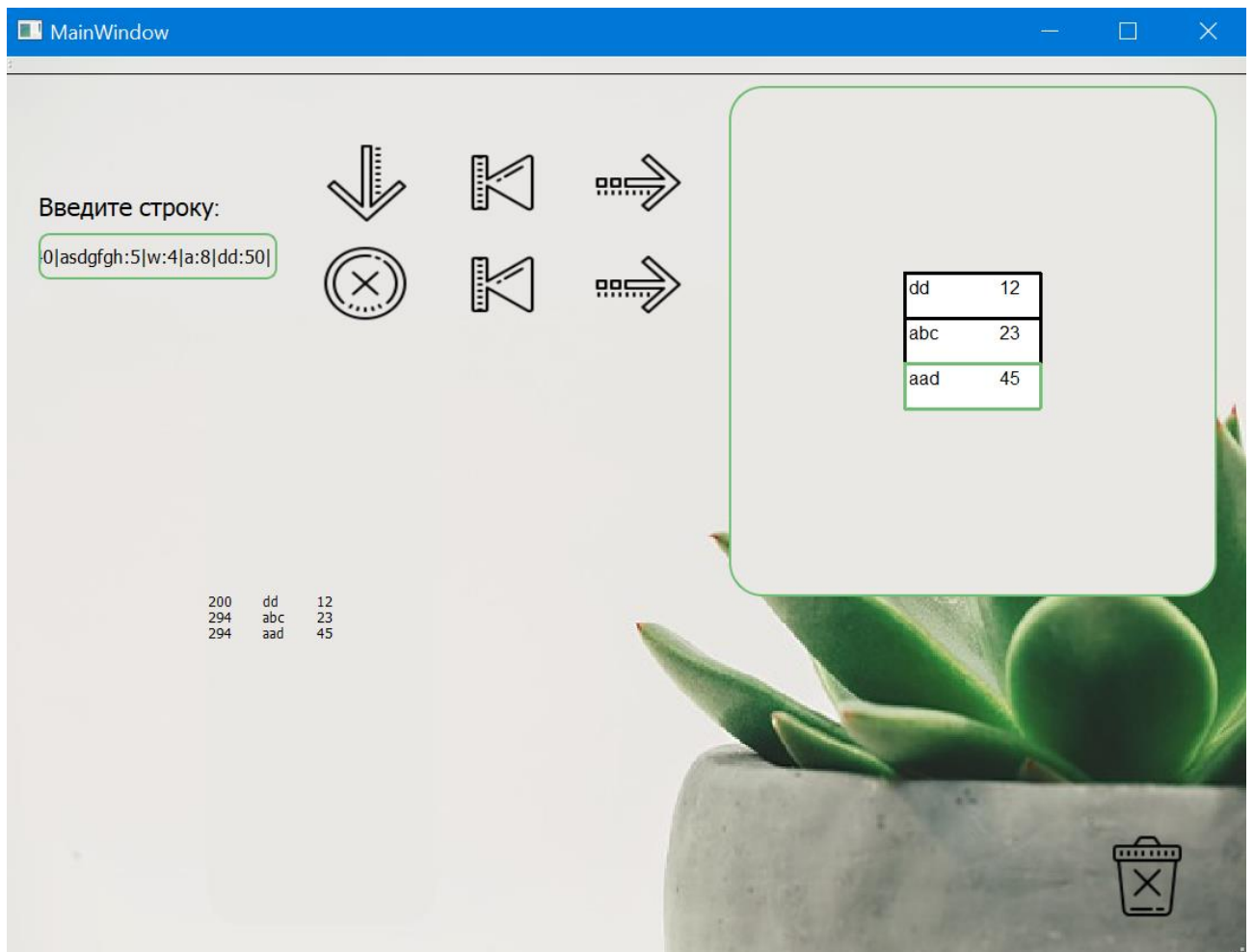
Рисунок 6 – Вид программы после очищения хеш-таблицы

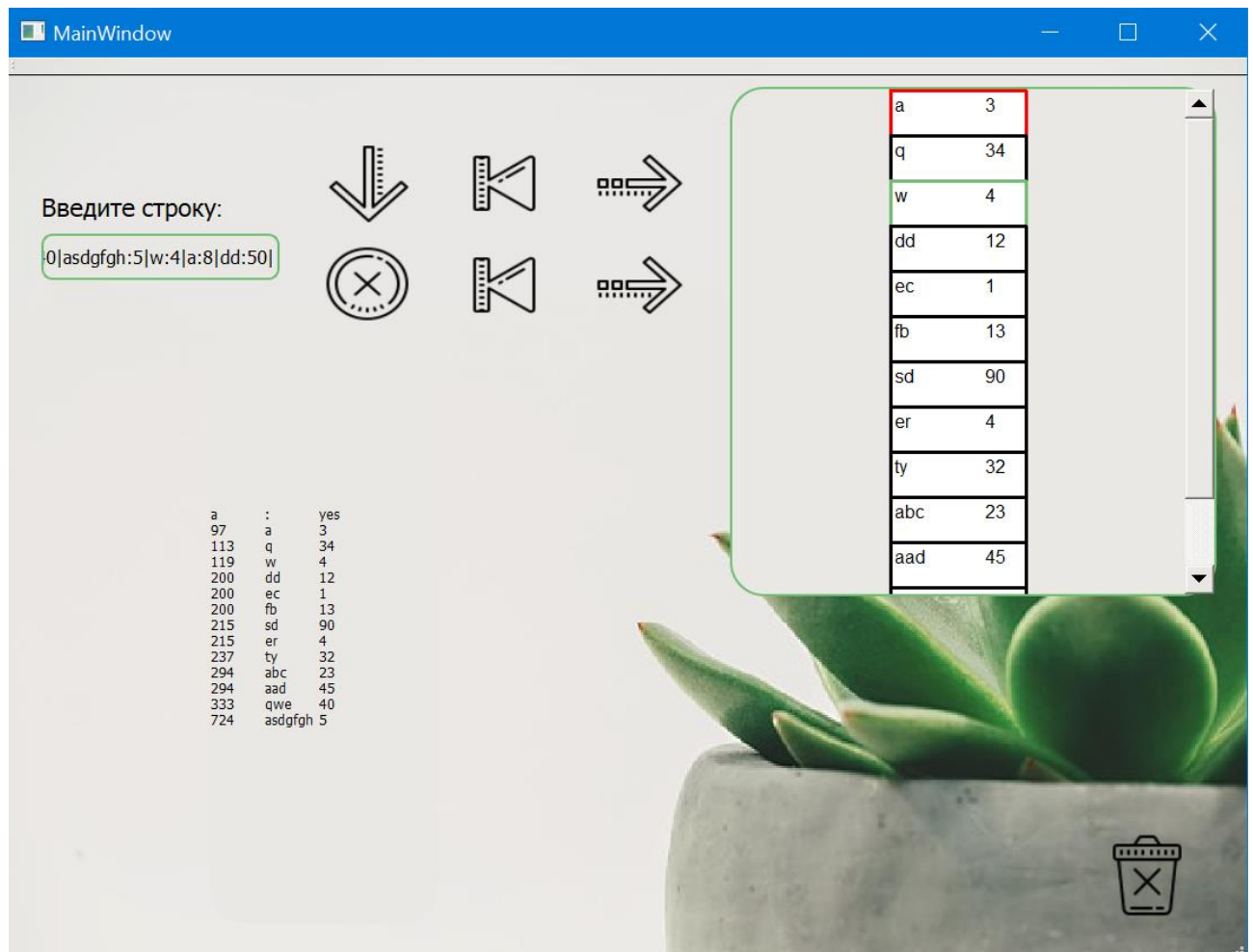
4. ДЕМОНСТРАЦИЯ

4.1. Демонстрация работы вставки в хеш-таблицу шаг вперед.









4.2. Демонстрация работы вставки в хеш-таблицу шаг назад.


MainWindow

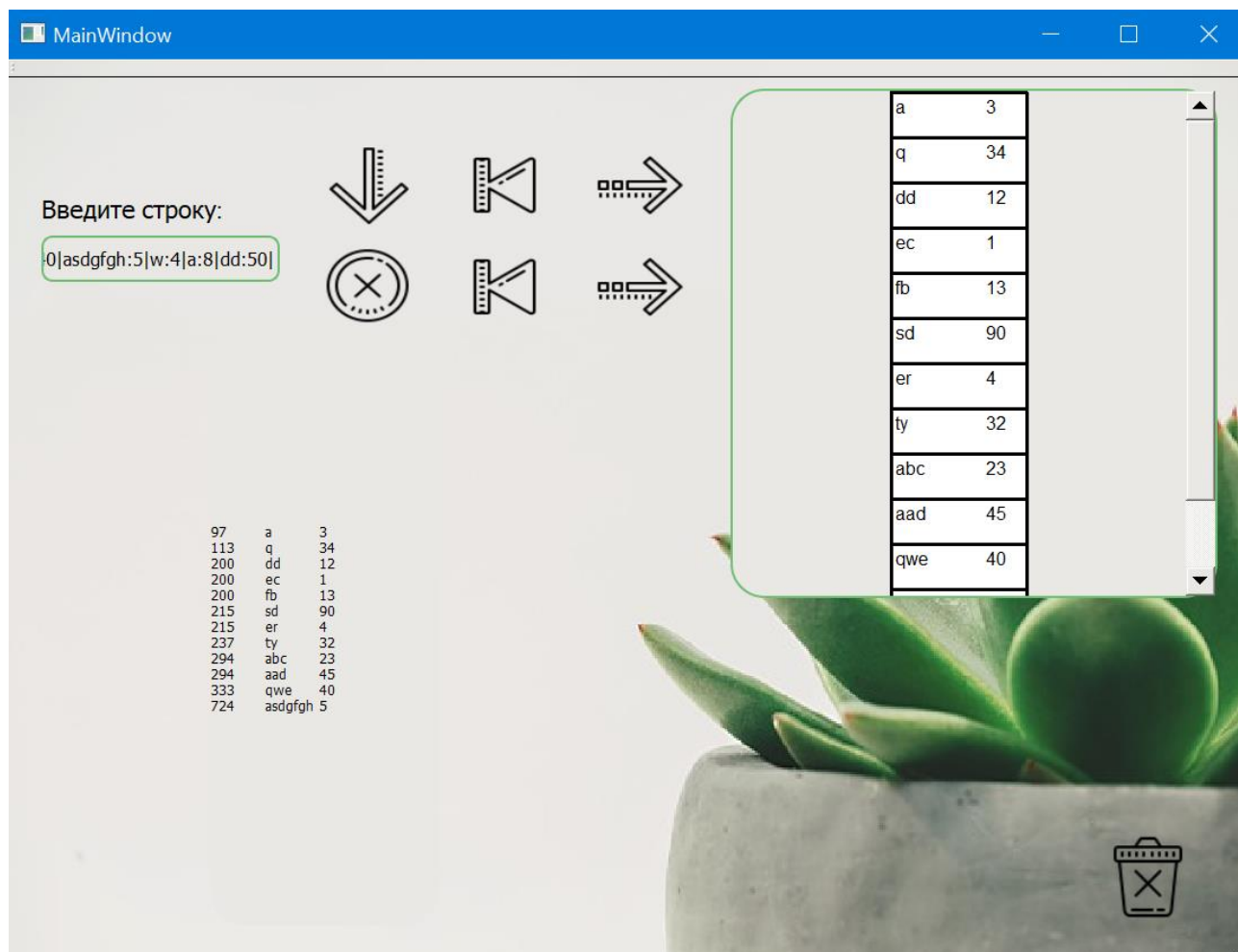
Введите строку:

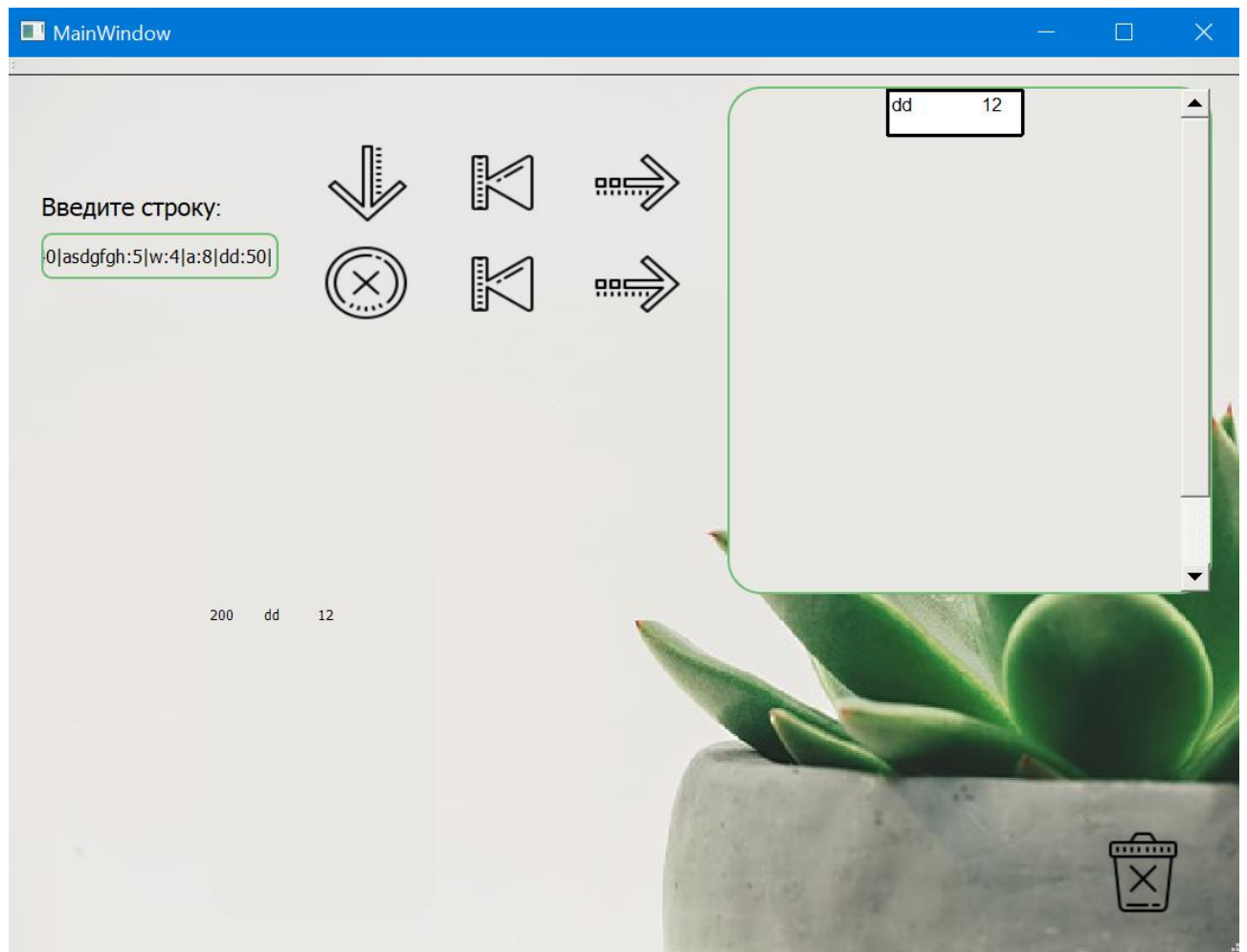
0|asdgh:5|w:4|a:8|dd:50|

97 a 3
113 q 34
119 w 4
200 dd 12
200 ec 1
200 fb 13
215 sd 90
215 er 4
237 ty 32
294 abc 23
294 aad 45
333 qwe 40
724 asdgh 5

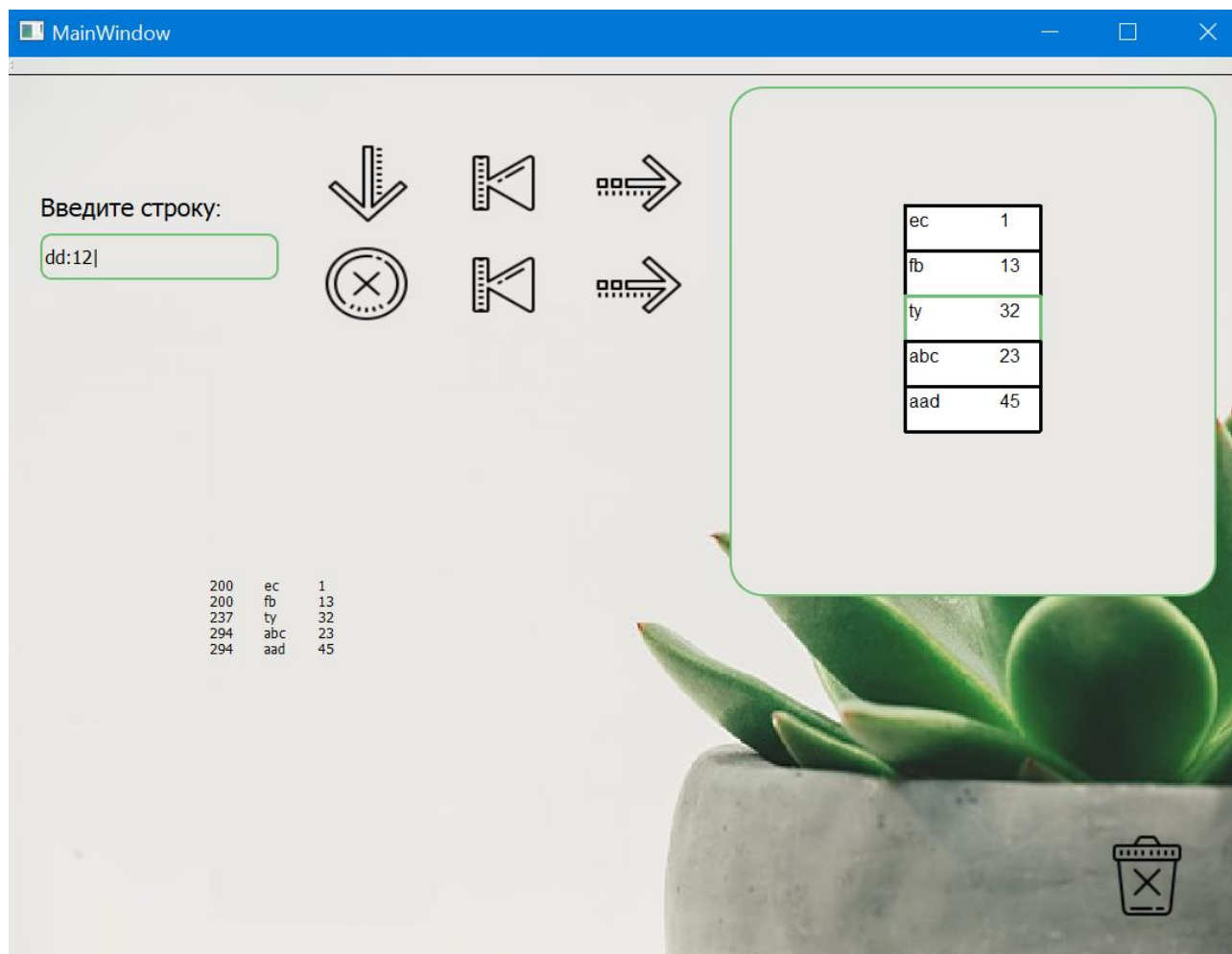
a	3
q	34
w	4
dd	12
ec	1
fb	13
sd	90
er	4
ty	32
abc	23
aad	45

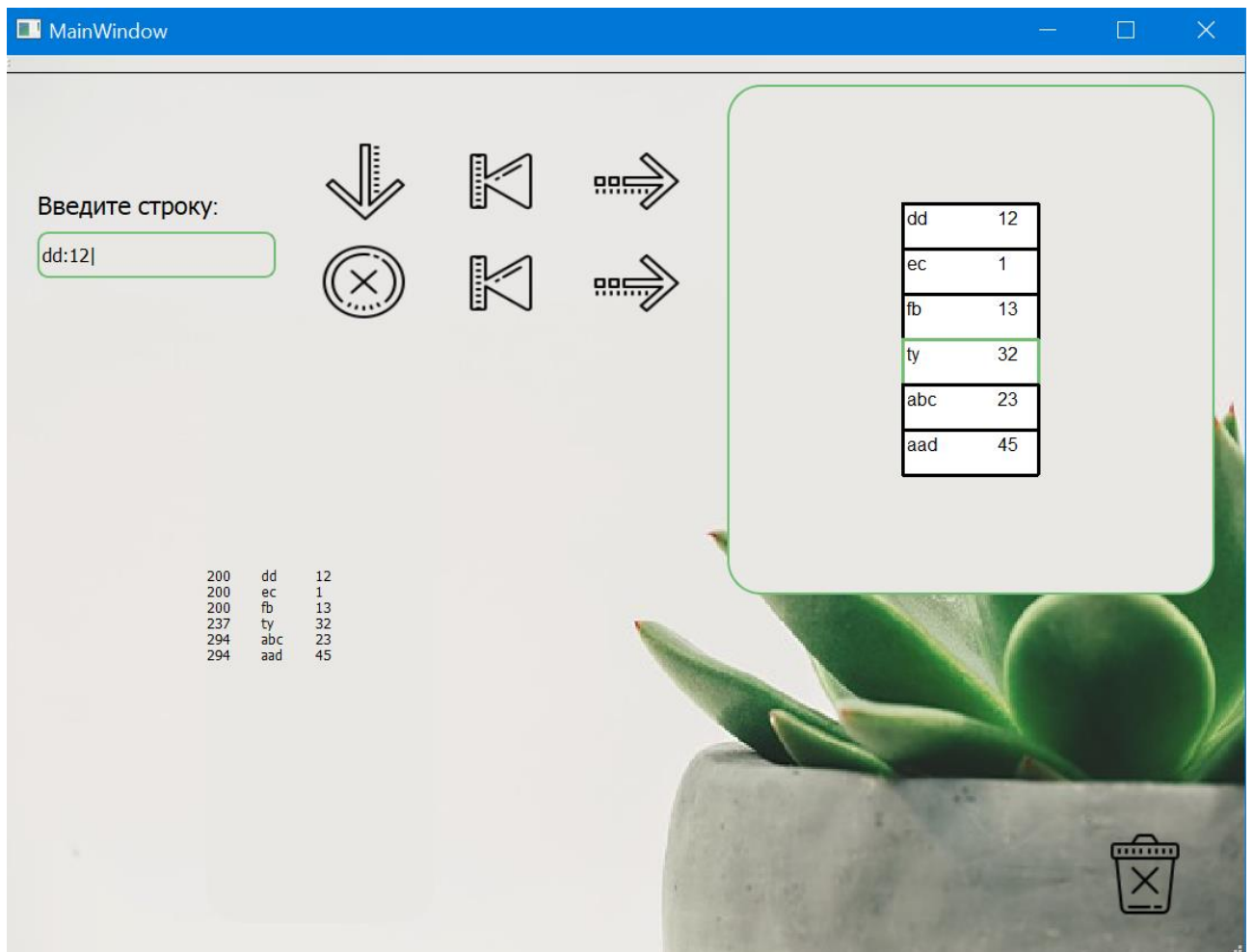


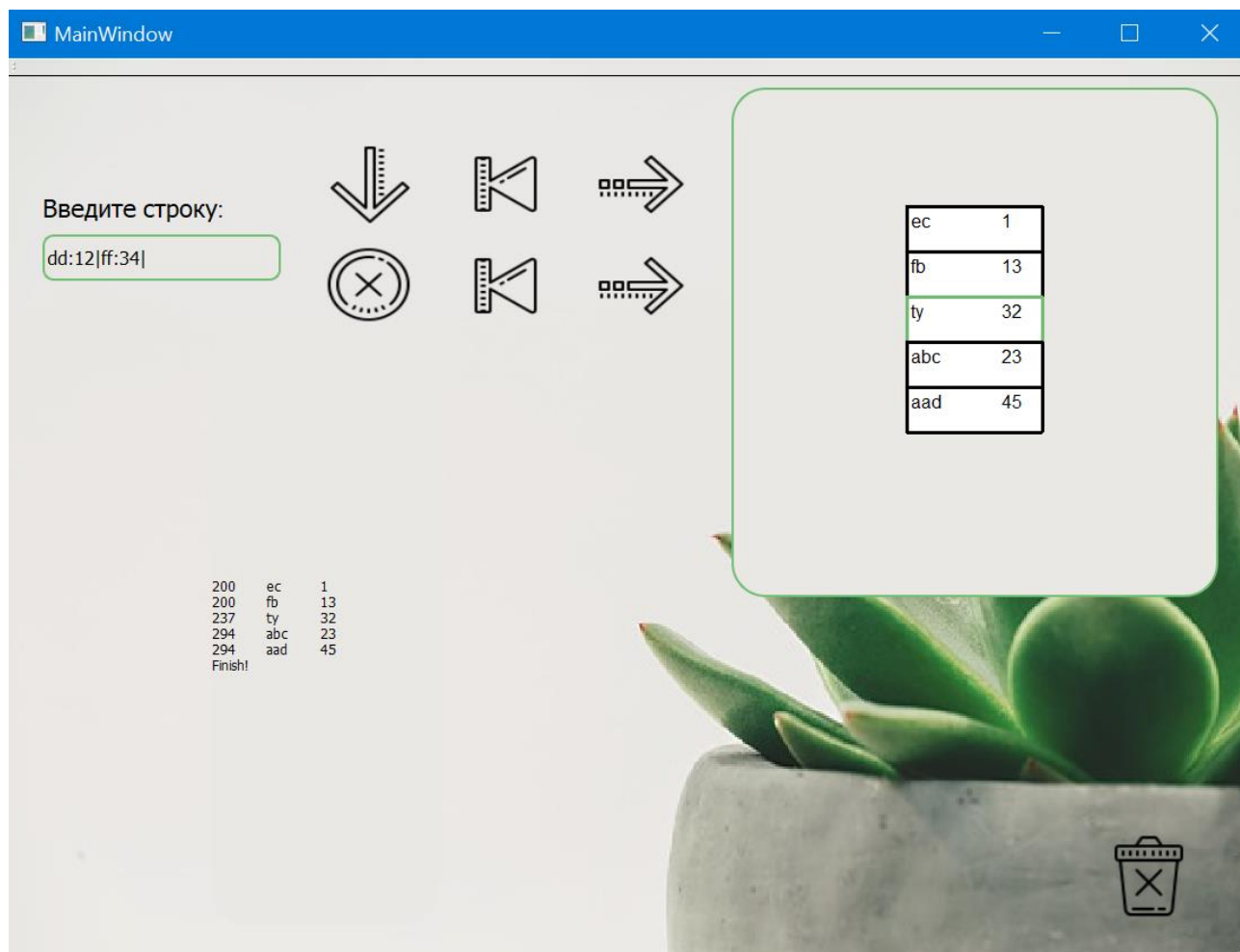




4.2. Демонстрация работы исключения из хеш-таблицы шаг вперед и шаг назад.







ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана программа, которая обладает следующей функциональностью: создание хеш-таблицы по хеш-функции, добавление и исключение элементов, шаг вперед, шаг назад, моментальный вывод хеш-таблицы, демонстрация рисунка, как выглядит хеш-таблица в данный момент.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Bjarne Stroustrup. A Tour of C++. М.: Addison-Wesley, 2018. 217 с.
2. Макс Шлее. Qt5.10. Профессиональное программирование на C++. М.: BHV-СПб, 2018, 513 с.
3. Перевод и дополнение документации QT // CrossPlatform.RU. URL: <http://doc.crossplatform.ru/>
4. Qt Documentation // Qt. URL: <https://doc.qt.io/qt-5/index.html>
5. Documentation // QCustomPlot. URL: <https://www.qcustomplot.com/index.php/support/documentation>
6. Хеш-таблицы // AlgoList. URL: http://algotlist.ru/ds/s_has.php
7. Примеры хеш-функций // Poznayka. URL: <https://poznayka.org/s97484t1.html>

ПРИЛОЖЕНИЕ А.
ИСХОДНЫЙ КОД ПРОГРАММЫ. MAIN.C

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();
    return a.exec();
}
```


ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД ПРОГРАММЫ. FUNCTIONS.H

```
#include <iostream>
#include <string>
#include <cctype>
#include <string>
#include <fstream>
#include "h_class.h"

using namespace std;

void open(string &str);
int sum(string strNew);
void make_str(Hash_cl<string, string> &Hash_, string &str, string &output,int
oper);
void make_str_step(Hash_cl<string, string> &Hash_st, string &str_step, string
&output,int &ind_st, int oper);
void make_str_del(Hash_cl<string, string> &Hash_st, string &str_step, string
&output,int &ind_st, int oper);
```

ПРИЛОЖЕНИЕ В

ИСХОДНЫЙ КОД ПРОГРАММЫ. FUNCTIONS.CPP

```
#include "functions.h"

void make_str(Hash_cl<string, string> &Hash_, string &str, string &output, int
oper){
    string ke = "";
    string val = "";
    int flag = 0;
    int su;
    for (unsigned int i = 0; i < str.length(); i++) {
        if (str[i] == '|') {
            su = sum(ke);
            if (!oper){
                Hash_.do_hash_str(ke, val, su, output);
            }
            else {
                Hash_.delete_elem(ke, val, su);
            }
            flag = 0;
            ke = "";
            val = "";
            continue;
        }
        if (str[i] == ':') {
            flag = 1;
            continue;
        }
        if (flag == 0){
            ke += str[i];
        } else
            val += str[i];
    }
}

void make_str_step(Hash_cl<string, string> &Hash_st, string &str_step, string
&output, int &ind_st, int oper){
    string ke = "";
    string val = "";
    int flag = 0;
    int su;
    //static unsigned int ind = 0;
    while (str_step[ind_st] != '|' && str_step[ind_st] != '\0'){
        if (str_step[ind_st] == ':') {
            flag = 1;
            ind_st++;
            continue;
        }
    }
}
```

```

    }
    if (flag == 0){
        ke += str_step[ind_st];
    } else
        val += str_step[ind_st];
    ind_st++;
}
su = sum(ke);
if (!oper){
    Hash_st.do_hash_str(ke, val, su, output);
}
else {
    Hash_st.delete_elem(ke, val, su);
}
ind_st++;
}

void make_str_del(Hash_cl<string, string> &Hash_st, string &str_step, string
&output,int &ind_st, int oper){
    string ke = "";
    string val = "";
    int flag = 0;
    int su;
    if (str_step[ind_st-1] == '|')
        ind_st = ind_st - 2;
    while (str_step[ind_st] != '|' && ind_st!=0){
        ind_st--;
    }
    if (ind_st!=0)
        ind_st++;
    while (str_step[ind_st] != '|' && str_step[ind_st] != '\\0'){
        if (str_step[ind_st] == ':') {
            flag = 1;
            ind_st++;
            continue;
        }
        if (flag == 0){
            ke += str_step[ind_st];
        } else
            val += str_step[ind_st];
        ind_st++;
    }
    ind_st--;
    while (str_step[ind_st] != '|' && ind_st!=0){
        ind_st--;
    }
    if (ind_st!=0)
        ind_st++;
    su = sum(ke);
    if (!oper){
        Hash_st.do_hash_del(ke, val, su);
    }
}

```

```

        else {
            Hash_st.reset_elem(ke, val, su);
        }
    }

void open(string &str) {
    ifstream fin; // создаем объект класса ifstream (считать)
    fin.open("D:\\prog\\cpp\\lab1\\text.txt"); // открываем файл для считывания
    fin >> str;
    if (!fin.is_open()) cout << "ERROR";
    fin.close(); // закрываем файл
}

int sum(string strNew) {
    int sum = 0;
    for (unsigned int i = 0; i < strNew.length(); i++) {
        sum += int(strNew[i]);
    }
    return sum;
}

```

ПРИЛОЖЕНИЕ Г

ИСХОДНЫЙ КОД ПРОГРАММЫ. H_CLASS.H

```
#include <iostream>
#include <string>
#include <cctype>
#include <string>
#include <fstream>
using namespace std;

template <class Key, class Value> class Hash_cl
{
    struct Hash {
        Key key1;
        Value value1;
        int h_func = 0;
        int flag_repeat = 0;
        int fl_del = 0;
    };

    int size = 100;
    Hash *val = new Hash[size];
    int last_added = -1;
    int last_repeat = -1;

public:

    int get_size() {
        return size;
    }

    int get_last_added() {
        return last_added;
    }

    int get_last_repeat() {
        return last_repeat;
    }

    void reset_last_repeat() {
        last_repeat = -1;
    }

    void delete_all(){
```

```

        for (int i = 0; i < size; i++) {
            val[i].key1 = "";
            val[i].value1 = "";
            val[i].h_func = 0;
            val[i].flag_repeat = 0;
            val[i].fl_del = 0;
        }
    }

void resize(int sum) {
    Hash *arr = new Hash[sum];
    for (int i = 0; i < size; i++) {
        arr[i] = val[i];
    }
    delete[] val;
    val = arr;
    size = sum;
}

void do_hash_str(Key key, Value value, int h_sum, string &output) {
    if (h_sum >= size){
        resize(h_sum+100);
    }
    if (val[h_sum].h_func == 0) {
        val[h_sum].key1 = key;
        val[h_sum].value1 = value;
        val[h_sum].h_func = h_sum;
        last_added = h_sum;
    } else {
        int i = h_sum;
        while (val[i].h_func != 0){
            if (val[i].key1 == key) {
                last_repeat = i;
                output += key;
                output += "\t:\tyes\n";
                if (val[i].key1 == key && val[i].value1 == value) {
                    val[i].flag_repeat += 1;
                }
                return;
            }
            i++;
        }
        val[i].key1 = key;
        val[i].value1 = value;
        val[i].h_func = h_sum;
    }
}

```

```

        last_added = i;
    }
}

void do_hash_del(Key key, Value value, int h_sum){
    int i = h_sum;
    while (i < size){
        if (val[i].key1 == key && val[i].value1 == value) {
            if(val[i].flag_repeat > 0){
                val[i].flag_repeat--;
                if (get_last_repeat() == i)
                    last_repeat = -1;
                return;
            }
            val[i].key1 = "";
            val[i].value1 = "";
            val[i].h_func = 0;
            if (get_last_repeat() == i)
                last_repeat = -1;
            return;
        }
        i++;
    }
}

void delete_elem(Key key, Value value, int h_sum){
    int i = h_sum;
    while (i < size){
        if (val[i].key1 == key && val[i].value1 == value) {
            val[i].fl_del += 1;
            return;
        }
        i++;
    }
}

void reset_elem(Key key, Value value, int h_sum){
    int i = h_sum;
    while (i < size){
        if (val[i].key1 == key && val[i].value1 == value) {
            val[i].fl_del -= 1;
            return;
        }
        i++;
    }
}

```

```

void print_hash(string &output) {
    for (int i = 0; i < size; i++) {
        if (val[i].h_func != 0 && val[i].fl_del == 0) {
            output += to_string(val[i].h_func);
            output += "\t";
            output += val[i].key1;
            output += "\t";
            output += val[i].value1;
            output += "\n";
        }
    }
}

Hash operator[](unsigned int index){
    return val[index];
}

};

```


ПРИЛОЖЕНИЕ Д

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.H

```
#include <QMainWindow>
#include "functions.h"
#pragma once
#include "printhash.h"
#include <fstream>
#include <QMessageBox>
#include <QGraphicsScene>

namespace Ui {
class MainWindow;
}

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
    void on_Push_File_clicked();

    void on_nextStep_clicked();

    void on_prevStep_clicked();

    void on_del_elem_clicked();

    void on_prev_del_clicked();

    void on_next_del_clicked();

    void on_del_all_clicked();

private:
    Ui::MainWindow *ui;
    QGraphicsScene *scene;
    Hash_cl<string, string> Hash_;
    Hash_cl<string, string> Hash_st;
    string str;
    string str_step;
    string str_del = "";
    string str_del_step;
    int ind_st = 0;
};
```

ПРИЛОЖЕНИЕ Е

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.H.CPP

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    //ui->output->setWordWrap(true);
    QPixmap bkgnd("D:\\\\prog\\\\Styles\\\\plant-2004483_640.jpg");
    bkgnd = bkgnd.scaled(this->size(), Qt::IgnoreAspectRatio);
    QPalette palette;
    palette.setBrush(QPalette::Background, bkgnd);
    this->setPalette(palette);
    scene = new QGraphicsScene;
    ui->graphicsView->setScene(scene);
    open(str);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_Push_File_clicked()
{
    string output = "";
    str += qPrintable(ui->input->text());
    make_str(Hash_, str, output, 0);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    str = "";
    graphic(Hash_, scene);
}

void MainWindow::on_nextStep_clicked()
{
    ui->prevStep->setEnabled(true);
    open(str_step);
    string output = "";
    str_step += qPrintable(ui->input->text());
    if (ind_st == str_step.length()){
```

```

        Hash_st.print_hash(output);
        output += "Finish!";
        ui->output->setText(QString::fromStdString(output));
        ui->nextStep->setEnabled(false);
        return;
    }
    make_str_step(Hash_st, str_step, output, ind_st, 0);
    Hash_st.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_st, scene);
}

void MainWindow::on_prevStep_clicked()
{
    ui->nextStep->setEnabled(true);
    open(str_step);
    string output = "";
    str_step += qPrintable(ui->input->text());
    if (ind_st == 0){
        Hash_st.print_hash(output);
        output += "No elements!";
        ui->output->setText(QString::fromStdString(output));
        ui->prevStep->setEnabled(false);
        return;
    }
    make_str_del(Hash_st, str_step, output, ind_st, 0);
    Hash_st.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_st, scene);
}

void MainWindow::on_del_elem_clicked()
{
    string output = "";
    str_del += qPrintable(ui->input->text());
    make_str(Hash_, str_del, output, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
    str_del = "";
}

void MainWindow::on_prev_del_clicked()
{
    ui->next_del->setEnabled(true);
    string output = "";

```

```

    str_del_step += qPrintable(ui->input->text());
    if (ind_st == 0){
        Hash_.print_hash(output);
        output += "No elements!";
        ui->output->setText(QString::fromStdString(output));
        ui->prev_del->setEnabled(false);
        return;
    }
    make_str_del(Hash_, str_del_step, output, ind_st, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
    str_del_step = "";
}

void MainWindow::on_next_del_clicked()
{
    ui->prev_del->setEnabled(true);
    string output = "";
    str_del_step += qPrintable(ui->input->text());
    if (ind_st == str_del_step.length()){
        Hash_.print_hash(output);
        output += "Finish!";
        ui->output->setText(QString::fromStdString(output));
        ui->next_del->setEnabled(false);
        return;
    }
    make_str_step(Hash_, str_del_step, output, ind_st, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
    str_del_step = "";
}

void MainWindow::on_del_all_clicked()
{
    ind_st = 0;
    str = "";
    open(str);
    string output = "";
    Hash_.delete_all();
    Hash_st.delete_all();
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
}

```

ПРИЛОЖЕНИЕ Ж

ИСХОДНЫЙ КОД ПРОГРАММЫ. PRINTHASH.H

```
#include "mainwindow.h"
#include "QGraphicsScene"
#include "QGraphicsTextItem"
#include <QPen>
#include <QFont>

//int size_el = 10000;

QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene *&scene);
void add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1, int
y1, int x2, int y2, QPen &pen, QBrush &brush, QFont &font);
```

ПРИЛОЖЕНИЕ И

ИСХОДНЫЙ КОД ПРОГРАММЫ. PRINTHASH.CPP

```
#include "printhash.h"

QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene
*&scene){
    scene->clear();
    QPen pen;
    QColor color;
    color.setRgb(0, 0, 0);
    pen.setColor(color);
    pen.setCapStyle(Qt::RoundCap);
    pen.setJoinStyle(Qt::RoundJoin);
    QBrush brush (QColor::fromRgb(255,255,255));
    QFont font;
    font.setFamily("Arial");
    pen.setWidth(3);
    add_rect(scene, Hash_, 20, 20, 120, 40, pen, brush, font); //x1, y1, h, w
}

void add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1,
int y1, int h, int w, QPen &pen, QBrush &brush, QFont &font){
    QString out;
    for (int i = 0; i < Hash_.get_size(); i++) {
        if (Hash_[i].h_func != 0 && Hash_[i].fl_del == 0){
            out.clear();
            out += QString::fromStdString(Hash_[i].key1);
            out += "\\t";
            out += QString::fromStdString(Hash_[i].value1);
            QGraphicsTextItem *textItem = new QGraphicsTextItem;
            textItem->setPos(x1, y1);
            textItem->setPlainText(out);
            textItem->setFont(font);
            if (i == Hash_.get_last_added())
                pen.setColor(QColor::fromRgb(114,190,116));
            if (i == Hash_.get_last_repeat()){
                pen.setColor(QColor::fromRgb(255,0,0));
                Hash_.reset_last_repeat();
            }
            scene->addRect(x1, y1, h, w, pen, brush);
            scene->addItem(textItem);
            pen.setColor(QColor::fromRgb(0,0,0));
            y1 += 40;
        }
    }
    return;
}
```

ПРИЛОЖЕНИЕ К

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.H

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H
#include <QMainWindow>
#include "strstrtester.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:

    void on_GPushOk_clicked();

    void on_HPushOk_clicked();

    void on_MPushOk_clicked();

    void on_OutSetPlot_toggled(bool checked);

    void on_OutSaveFile_clicked();

private:
    void setupPlot();
    Ui::MainWindow *ui;
    StrStrWorker ssw;
};
#endif // MAINWINDOW_H
```

ПРИЛОЖЕНИЕ Л

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.CPP

```
#include "printhash.h"

QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene
*&scene){
    scene->clear();
    QPen pen;
    QColor color;
    color.setRgb(0, 0, 0);
    pen.setColor(color);
    pen.setCapStyle(Qt::RoundCap);
    pen.setJoinStyle(Qt::RoundJoin);
    QBrush brush (QColor::fromRgb(255,255,255));
    QFont font;
    font.setFamily("Arial");
    pen.setWidth(3);
    add_rect(scene, Hash_, 20, 20, 120, 40, pen, brush, font); //x1, y1, h, w
}

void add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1,
int y1, int h, int w, QPen &pen, QBrush &brush, QFont &font){
    QString out;
    for (int i = 0; i < Hash_.get_size(); i++) {
        if (Hash_[i].h_func != 0 && Hash_[i].fl_del == 0){
            out.clear();
            out += QString::fromStdString(Hash_[i].key1);
            out += "\\t";
            out += QString::fromStdString(Hash_[i].value1);
            QGraphicsTextItem *textItem = new QGraphicsTextItem;
            textItem->setPos(x1, y1);
            textItem->setPlainText(out);
            textItem->setFont(font);
            if (i == Hash_.get_last_added())
                pen.setColor(QColor::fromRgb(114,190,116));
            if (i == Hash_.get_last_repeat()){
                pen.setColor(QColor::fromRgb(255,0,0));
                Hash_.reset_last_repeat();
            }
            scene->addRect(x1, y1, h, w, pen, brush);
            scene->addItem(textItem);
            pen.setColor(QColor::fromRgb(0,0,0));
            y1 += 40;
        }
    }
    return;
}

#include "printhash.h"
```



```

QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene
*&scene){
    scene->clear();
    QPen pen;
    QColor color;
    color.setRgb(0, 0, 0);
    pen.setColor(color);
    pen.setCapStyle(Qt::RoundCap);
    pen.setJoinStyle(Qt::RoundJoin);
    QBrush brush (QColor::fromRgb(255,255,255));
    QFont font;
    font.setFamily("Arial");
    pen.setWidth(3);
    add_rect(scene, Hash_, 20, 20, 120, 40, pen, brush, font); //x1, y1, h, w
}

```

```

void add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1,
int y1, int h, int w, QPen &pen, QBrush &brush, QFont &font){
    QString out;
    for (int i = 0; i < Hash_.get_size(); i++) {
        if (Hash_[i].h_func != 0 && Hash_[i].fl_del == 0){
            out.clear();
            out += QString::fromStdString(Hash_[i].key1);
            out += "\t";
            out += QString::fromStdString(Hash_[i].value1);
            QGraphicsTextItem *textItem = new QGraphicsTextItem;
            textItem->setPos(x1, y1);
            textItem->setPlainText(out);
            textItem->setFont(font);
            if (i == Hash_.get_last_added())
                pen.setColor(QColor::fromRgb(114,190,116));
            if (i == Hash_.get_last_repeat()){
                pen.setColor(QColor::fromRgb(255,0,0));
                Hash_.reset_last_repeat();
            }
            scene->addRect(x1, y1, h, w, pen, brush);
            scene->addItem(textItem);
            pen.setColor(QColor::fromRgb(0,0,0));
            y1 += 40;
        }
    }
    return;
}

```

```

#include "printhash.h"

```

```

QGraphicsScene *graphic(Hash_cl<string, string> &Hash_, QGraphicsScene
*&scene){
    scene->clear();
    QPen pen;
    QColor color;
    color.setRgb(0, 0, 0);

```

```

        pen.setColor(color);
        pen.setCapStyle(Qt::RoundCap);
        pen.setJoinStyle(Qt::RoundJoin);
        QBrush brush (QColor::fromRgb(255,255,255));
        QFont font;
        font.setFamily("Arial");
        pen.setWidth(3);
        add_rect(scene, Hash_, 20, 20, 120, 40, pen, brush, font); //x1, y1, h, w
    }

void add_rect(QGraphicsScene *&scene, Hash_cl<string, string> &Hash_, int x1,
int y1, int h, int w, QPen &pen, QBrush &brush, QFont &font){
    QString out;
    for (int i = 0; i < Hash_.get_size(); i++) {
        if (Hash_[i].h_func != 0 && Hash_[i].fl_del == 0){
            out.clear();
            out += QString::fromStdString(Hash_[i].key1);
            out += "\\t";
            out += QString::fromStdString(Hash_[i].value1);
            QGraphicsTextItem *textItem = new QGraphicsTextItem;
            textItem->setPos(x1, y1);
            textItem->setPlainText(out);
            textItem->setFont(font);
            if (i == Hash_.get_last_added())
                pen.setColor(QColor::fromRgb(114,190,116));
            if (i == Hash_.get_last_repeat()){
                pen.setColor(QColor::fromRgb(255,0,0));
                Hash_.reset_last_repeat();
            }
            scene->addRect(x1, y1, h, w, pen, brush);
            scene->addItem(textItem);
            pen.setColor(QColor::fromRgb(0,0,0));
            y1 += 40;
        }
    }
    return;
}

```

```

#include "mainwindow.h"
#include "ui_mainwindow.h"

```

```

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    //ui->output->setWordWrap(true);
    QPixmap bkgnd("D:\\prog\\Styles\\plant-2004483_640.jpg");
    bkgnd = bkgnd.scaled(this->size(), Qt::IgnoreAspectRatio);
    QPalette palette;
    palette.setBrush(QPalette::Background, bkgnd);
    this->setPalette(palette);
}

```

```

        scene = new QGraphicsScene;
        ui->graphicsView->setScene(scene);
        open(str);
    }

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_Push_File_clicked()
{
    string output = "";
    str += qPrintable(ui->input->text());
    make_str(Hash_, str, output, 0);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    str = "";
    graphic(Hash_, scene);
}

void MainWindow::on_nextStep_clicked()
{
    ui->prevStep->setEnabled(true);
    open(str_step);
    string output = "";
    str_step += qPrintable(ui->input->text());
    if (ind_st == str_step.length()){
        Hash_st.print_hash(output);
        output += "Finish!";
        ui->output->setText(QString::fromStdString(output));
        ui->nextStep->setEnabled(false);
        return;
    }
    make_str_step(Hash_st, str_step, output, ind_st, 0);
    Hash_st.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_st, scene);
}

void MainWindow::on_prevStep_clicked()
{
    ui->nextStep->setEnabled(true);
    open(str_step);
    string output = "";
    str_step += qPrintable(ui->input->text());
    if (ind_st == 0){
        Hash_st.print_hash(output);
        output += "No elements!";
        ui->output->setText(QString::fromStdString(output));
        ui->prevStep->setEnabled(false);
        return;
    }
}

```

```

    }
    make_str_del(Hash_st, str_step, output, ind_st, 0);
    Hash_st.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_st, scene);
}

void MainWindow::on_del_elem_clicked()
{
    string output = "";
    str_del += qPrintable(ui->input->text());
    make_str(Hash_, str_del, output, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
    str_del = "";
}

void MainWindow::on_prev_del_clicked()
{
    ui->next_del->setEnabled(true);
    string output = "";
    str_del_step += qPrintable(ui->input->text());
    if (ind_st == 0){
        Hash_.print_hash(output);
        output += "No elements!";
        ui->output->setText(QString::fromStdString(output));
        ui->prev_del->setEnabled(false);
        return;
    }
    make_str_del(Hash_, str_del_step, output, ind_st, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
    str_del_step = "";
}

void MainWindow::on_next_del_clicked()
{
    ui->prev_del->setEnabled(true);
    string output = "";
    str_del_step += qPrintable(ui->input->text());
    if (ind_st == str_del_step.length()){
        Hash_.print_hash(output);
        output += "Finish!";
        ui->output->setText(QString::fromStdString(output));
        ui->next_del->setEnabled(false);
        return;
    }
    make_str_step(Hash_, str_del_step, output, ind_st, 1);
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
}

```

```

    str_del_step = "";
}

void MainWindow::on_del_all_clicked()
{
    ind_st = 0;
    str = "";
    open(str);
    string output = "";
    Hash_.delete_all();
    Hash_st.delete_all();
    Hash_.print_hash(output);
    ui->output->setText(QString::fromStdString(output));
    graphic(Hash_, scene);
}

```

ПРИЛОЖЕНИЕ М

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.UI

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>MainWindow</class>
  <widget class="QMainWindow" name="MainWindow">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>1098</width>
        <height>800</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>MainWindow</string>
    </property>
    <widget class="QWidget" name="centralWidget">
      <widget class="QLabel" name="label">
        <property name="geometry">
          <rect>
            <x>30</x>
            <y>100</y>
            <width>201</width>
            <height>31</height>
          </rect>
        </property>
        <property name="font">
          <font>
            <family>Ebrima</family>
            <pointsize>11</pointsize>
          </font>
        </property>
        <property name="text">
          <string>Введите строку:</string>
        </property>
      </widget>
      <widget class="QPushButton" name="Push_File">
        <property name="geometry">
          <rect>
            <x>280</x>
            <y>60</y>
            <width>80</width>
            <height>71</height>
          </rect>
        </property>
      </widget>
    </widget>
  </widget>
</ui>
```

```

    </rect>
  </property>
  <property name="cursor">
    <cursorShape>PointingHandCursor</cursorShape>
  </property>
  <property name="toolTip">
    <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;      -qt-block-indent:0;      text-indent:0px;      background-
color:#e9e8e4;&quot;&gt;&lt;span      style=&quot;      background-
color:#e9e8e4;&quot;&gt;Вывести
сrazy&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
  </property>
  <property name="whatsThis">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;br/&gt;&lt;/p&gt;&lt;
/body&gt;&lt;/html&gt;</string>
  </property>
  <property name="styleSheet">
    <string notr="true">border-image: url(:/C:/Users/Олеся/Downloads/icons8-
down-arrow-64.png)</string>
  </property>
  <property name="text">
    <string/>
  </property>
</widget>
<widget class="QLineEdit" name="input">
  <property name="geometry">
    <rect>
      <x>30</x>
      <y>140</y>
      <width>211</width>
      <height>41</height>
    </rect>
  </property>
  <property name="toolTip">
    <string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span style=&quot;
color:#008000;&quot;&gt;Format:      &lt;/span&gt;&lt;span style=&quot;
color:#008000;      background-

```

```

color:#e9e8e4;">(?:?|)*</span></p></body></html>
</string>
  </property>
  <property name="styleSheet">
    <string notr="true">border-radius: 10%;
border: 2px solid #72be74;
background-color: #e9e8e4;</string>
  </property>
</widget>
<widget class="QGraphicsView" name="graphicsView">
  <property name="geometry">
    <rect>
      <x>640</x>
      <y>10</y>
      <width>431</width>
      <height>451</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">border: 2px solid #72be74;
border-radius: 30%;
background-color: #e9e8e4;</string>
  </property>
</widget>
<widget class="QLabel" name="output">
  <property name="geometry">
    <rect>
      <x>180</x>
      <y>220</y>
      <width>201</width>
      <height>531</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 6pt "MS Shell Dlg 2";
border-radius: 30%;
background-color: #e9e8e4;
</string>
  </property>
  <property name="text">
    <string/>
  </property>
</widget>
<widget class="QPushButton" name="nextStep">
  <property name="geometry">
    <rect>

```



```

        <x>520</x>
        <y>60</y>
        <width>80</width>
        <height>71</height>
    </rect>
</property>
<property name="cursor">
    <cursorShape>PointingHandCursor</cursorShape>
</property>
<property name="mouseTracking">
    <bool>>false</bool>
</property>
<property name="toolTip">
    <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px; -qt-block-indent:0; text-indent:0px; background-
color:#e9e8e4;&quot;&gt;&lt;span
style=&quot; background-
color:#e9e8e4;&quot;&gt;&lt;var
вперед&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
<property name="styleSheet">
    <string notr="true">border-image: url(:/C:/Users/Олеся/Downloads/icons8-
advance-64.png)</string>
</property>
<property name="text">
    <string/>
</property>
</widget>
<widget class="QPushButton" name="prevStep">
    <property name="geometry">
        <rect>
            <x>400</x>
            <y>60</y>
            <width>80</width>
            <height>71</height>
        </rect>
    </property>
    <property name="cursor">
        <cursorShape>PointingHandCursor</cursorShape>
    </property>

```

```

        <property name="toolTip">
            <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;          -qt-block-indent:0;          text-indent:0px;          background-
color:#e9e8e4;&quot;&gt;&lt;span
style=&quot;          background-
color:#e9e8e4;&quot;&gt;¶¶¶
назад&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
        </property>
        <property name="styleSheet">
            <string      notr="true">border-image:url(:/C:/Users/Олеся/Downloads/icons8-
skip-to-start-64.png)</string>
        </property>
        <property name="text">
            <string/>
        </property>
    </widget>
    <widget class="QPushButton" name="del_elem">
        <property name="geometry">
            <rect>
                <x>280</x>
                <y>150</y>
                <width>80</width>
                <height>71</height>
            </rect>
        </property>
        <property name="cursor">
            <cursorShape>PointingHandCursor</cursorShape>
        </property>
        <property name="toolTip">
            <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;          -qt-block-indent:0;          text-indent:0px;          background-
color:#e9e8e4;&quot;&gt;&lt;span
style=&quot;          background-

```

```

color:#e9e8e4;">Удалить
элементы</span></p></body></html></string>
    </property>
    <property name="styleSheet">
        <string      notr="true">border-image:url(/C:/Users/Олеся/Downloads/icons8-
delete-64.png)</string>
    </property>
    <property name="text">
        <string/>
    </property>
</widget>
<widget class="QPushButton" name="prev_del">
    <property name="geometry">
        <rect>
            <x>400</x>
            <y>150</y>
            <width>80</width>
            <height>71</height>
        </rect>
    </property>
    <property name="cursor">
        <cursorShape>PointingHandCursor</cursorShape>
    </property>
    <property name="toolTip">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-/W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;      -qt-block-indent:0;      text-indent:0px;      background-
color:#e9e8e4;&quot;&gt;&lt;span      style=&quot;      background-
color:#e9e8e4;&quot;&gt;&lt;div
назад&lt;/span>&lt;/p>&lt;/body>&lt;/html>&lt;/string>
    </property>
    <property name="styleSheet">
        <string      notr="true">border-image:url(/C:/Users/Олеся/Downloads/icons8-
skip-to-start-64.png)</string>
    </property>
    <property name="text">
        <string/>
    </property>
</widget>
<widget class="QPushButton" name="next_del">

```

```

<property name="geometry">
  <rect>
    <x>520</x>
    <y>150</y>
    <width>80</width>
    <height>71</height>
  </rect>
</property>
<property name="cursor">
  <cursorShape>PointingHandCursor</cursorShape>
</property>
<property name="mouseTracking">
  <bool>>false</bool>
</property>
<property name="toolTip">
  <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot;font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;      -qt-block-indent:0;      text-indent:0px;      background-
color:#e9e8e4;&quot;&gt;&lt;span      style=&quot;      background-
color:#e9e8e4;&quot;&gt;War
вперед&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
  </property>
<property name="styleSheet">
  <string notr="true">border-image: url(:/C:/Users/Олеся/Downloads/icons8-
advance-64.png)</string>
</property>
<property name="text">
  <string/>
</property>
</widget>
<widget class="QPushButton" name="del_all">
  <property name="geometry">
    <rect>
      <x>969</x>
      <y>670</y>
      <width>81</width>
      <height>81</height>
    </rect>
  </property>
  <property name="cursor">

```

```

        <cursorShape>PointingHandCursor</cursorShape>
    </property>
    <property name="toolTip">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot; margin-top:12px; margin-bottom:12px; margin-left:0px; margin-
right:0px;          -qt-block-indent:0;          text-indent:0px;          background-
color:#e9e8e4;&quot;&gt;&lt;span                    style=&quot;          background-
color:#e9e8e4;&quot;&gt;Очистить
все&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
    <property name="styleSheet">
        <string      notr="true">border-image:url(/C:/Users/Олеся/Downloads/icons8-
delete-bin-64.png)</string>
    </property>
    <property name="text">
        <string/>
    </property>
</widget>
<zorder>label</zorder>
<zorder>Push_File</zorder>
<zorder>graphicsView</zorder>
<zorder>output</zorder>
<zorder>nextStep</zorder>
<zorder>prevStep</zorder>
<zorder>del_elem</zorder>
<zorder>prev_del</zorder>
<zorder>next_del</zorder>
<zorder>input</zorder>
<zorder>del_all</zorder>
</widget>
<widget class="QMenuBar" name="menuBar">
    <property name="geometry">
        <rect>
            <x>0</x>
            <y>0</y>
            <width>1098</width>
            <height>17</height>
        </rect>
    </property>
</widget>

```

```

<widget class="QToolBar" name="mainToolBar">
  <attribute name="toolBarArea">
    <enum>TopToolBarArea</enum>
  </attribute>
  <attribute name="toolBarBreak">
    <bool>false</bool>
  </attribute>
</widget>
<widget class="QStatusBar" name="statusBar"/>
</widget>
<layoutdefault spacing="6" margin="11"/>
<resources/>
<connections/>

</ui>

```

ПРИЛОЖЕНИЕ Н

ИСХОДНЫЙ КОД ПРОГРАММЫ. CW.PRO

```
#-----  
#  
# Project created by QtCreator 2019-12-11T19:30:50  
#  
#-----  
  
QT      += core gui  
  
greaterThan(QT_MAJOR_VERSION, 4): QT += widgets  
  
TARGET = cw  
TEMPLATE = app  
  
# The following define makes your compiler emit warnings if you use  
# any feature of Qt which has been marked as deprecated (the exact warnings  
# depend on your compiler). Please consult the documentation of the  
# deprecated API in order to know how to port your code away from it.  
DEFINES += QT_DEPRECATED_WARNINGS  
  
# You can also make your code fail to compile if you use deprecated APIs.  
# In order to do so, uncomment the following line.  
# You can also select to disable deprecated APIs only up to a certain version of  
# Qt.  
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs  
deprecated before Qt 6.0.0  
  
CONFIG += c++11  
  
SOURCES += \  
    functions.cpp \  
    main.cpp \  
    mainwindow.cpp \  
    printhash.cpp  
  
HEADERS += \  
    functions.h \  
    h_class.h \  
    mainwindow.h \  
    printhash.h  
  
FORMS += \  
    mainwindow.ui
```

```
# Default rules for deployment.
qnx: target.path = /tmp/${TARGET}/bin
else: unix:!android: target.path = /opt/${TARGET}/bin
!isEmpty(target.path): INSTALLS += target
```

```
RESOURCES += \
    pictures.qrc
```

ПРИЛОЖЕНИЕ О

ИСХОДНЫЙ КОД ПРОГРАММЫ. MAINWINDOW.UI

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
    <class>MainWindow</class>
    <widget class="QMainWindow" name="MainWindow">
        <property name="geometry">
            <rect>
                <x>0</x>
                <y>0</y>
                <width>1280</width>
                <height>1000</height>
            </rect>
        </property>
        <property name="windowTitle">
            <string>MainWindow</string>
        </property>
        <property name="styleSheet">
            <string notr="true">background-color: rgb(30, 30, 30)</string>
        </property>
        <widget class="QWidget" name="centralwidget">
            <widget class="QPushButton" name="GPushOk">
                <property name="geometry">
                    <rect>
                        <x>270</x>
                        <y>100</y>
                        <width>50</width>
                        <height>50</height>
                    </rect>
                </property>
                <property name="styleSheet">
                    <string notr="true">border-image: url(:/ok.png);</string>
                </property>
                <property name="text">
                    <string/>
                </property>
            </widget>
```



```

<widget class="QTextEdit" name="GInputPairs">
  <property name="geometry">
    <rect>
      <x>120</x>
      <y>50</y>
      <width>131</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt "Tahoma";
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
  </property>
  <property name="html">
    <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Tahoma'; font-
size:10pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
  </property>
</widget>
<widget class="QLabel" name="GNameMain">
  <property name="geometry">
    <rect>
      <x>20</x>
      <y>10</y>
      <width>331</width>
      <height>31</height>
    </rect>
  </property>
  <property name="font">
    <font>
      <family>Tahoma</family>
      <pointsize>11</pointsize>
      <weight>75</weight>
      <italic>false</italic>
      <bold>true</bold>

```

```

    </font>
</property>
<property name="layoutDirection">
    <enum>Qt::LeftToRight</enum>
</property>
<property name="styleSheet">
    <string notr="true">font: bold 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
</property>
<property name="text">
    <string>Generate file</string>
</property>
<property name="alignment">
    <set>Qt::AlignCenter</set>
</property>
</widget>
<widget class="QLabel" name="HNameMain">
    <property name="geometry">
        <rect>
            <x>360</x>
            <y>10</y>
            <width>611</width>
            <height>31</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">font: bold 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>Hash table file</string>
    </property>
    <property name="alignment">
        <set>Qt::AlignCenter</set>
    </property>
</widget>
<widget class="QPushButton" name="HPushOk">
    <property name="geometry">
        <rect>
            <x>920</x>
            <y>70</y>
            <width>50</width>
            <height>50</height>
        </rect>
    </property>
    <property name="styleSheet">

```

```

    <string notr="true">border-image: url(/ok.png);</string>
  </property>
  <property name="text">
    <string/>
  </property>
</widget>
<widget class="QTextEdit" name="GInputLength">
  <property name="geometry">
    <rect>
      <x>120</x>
      <y>105</y>
      <width>131</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt "Tahoma";
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
  </property>
  <property name="html">
    <string>&lt;!DOCTYPE HTML PUBLIC "http://www.w3.org/TR/REC-html40/strict.dtd"
&lt;http://www.w3.org/TR/REC-html40/strict.dtd"
&lt;html&gt;&lt;head&gt;&lt;meta name="qrichtext" content="1"
/&gt;&lt;style type="text/css&gt;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style="font-family:'Tahoma'; font-size:10pt; font-weight:400; font-style:normal;"&gt;
&lt;p style="-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;"&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
  </property>
</widget>
<widget class="QLabel" name="GNamePairs">
  <property name="geometry">
    <rect>
      <x>10</x>
      <y>50</y>
      <width>101</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">

```

```

        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>Pairs num:</string>
    </property>
    <property name="alignment">
        <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
</widget>
<widget class="QLabel" name="GNameLength">
    <property name="geometry">
        <rect>
            <x>10</x>
            <y>105</y>
            <width>101</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>Key length:</string>
    </property>
    <property name="alignment">
        <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
</widget>
<widget class="QLabel" name="GNameFile">
    <property name="geometry">
        <rect>
            <x>10</x>
            <y>160</y>
            <width>101</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>File name:</string>
    </property>

```

```

    <property name="alignment">
      <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
  </widget>
  <widget class="QTextEdit" name="GInputFile">
    <property name="geometry">
      <rect>
        <x>120</x>
        <y>160</y>
        <width>131</width>
        <height>41</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt &quot;Tahoma&quot;;
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
    </property>
    <property name="html">
      <string>&lt;!DOCTYPE HTML PUBLIC &quot;-/W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;richtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Tahoma'; font-
size:10pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
  </widget>
  <widget class="QLabel" name="HNameFile">
    <property name="geometry">
      <rect>
        <x>600</x>
        <y>50</y>
        <width>101</width>
        <height>41</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>

```

```

</property>
<property name="text">
  <string>Input file:</string>
</property>
<property name="alignment">
  <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
</property>
</widget>
<widget class="Line" name="VLine">
  <property name="geometry">
    <rect>
      <x>340</x>
      <y>25</y>
      <width>3</width>
      <height>310</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">background-color: rgb(70, 70, 70);</string>
  </property>
  <property name="orientation">
    <enum>Qt::Vertical</enum>
  </property>
</widget>
<widget class="Line" name="HLine">
  <property name="geometry">
    <rect>
      <x>365</x>
      <y>160</y>
      <width>600</width>
      <height>3</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">background-color: rgb(70, 70, 70);</string>
  </property>
  <property name="orientation">
    <enum>Qt::Horizontal</enum>
  </property>
</widget>
<widget class="QLabel" name="MNameMain">
  <property name="geometry">
    <rect>
      <x>360</x>
      <y>180</y>
      <width>611</width>

```

```

        <height>31</height>
    </rect>
</property>
<property name="styleSheet">
    <string notr="true">font: bold 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
</property>
<property name="text">
    <string>Graphmaker</string>
</property>
<property name="alignment">
    <set>Qt::AlignCenter</set>
</property>
</widget>
<widget class="QCheckBox" name="GCheckWorst">
    <property name="geometry">
        <rect>
            <x>60</x>
            <y>230</y>
            <width>191</width>
            <height>22</height>
        </rect>
    </property>
    <property name="layoutDirection">
        <enum>Qt::LeftToRight</enum>
    </property>
    <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>Worst case for HFunc</string>
    </property>
</widget>
<widget class="QLabel" name="MNameMin">
    <property name="geometry">
        <rect>
            <x>370</x>
            <y>255</y>
            <width>41</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>

```

```

    </property>
    <property name="text">
        <string>Min:</string>
    </property>
    <property name="alignment">
        <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
</widget>
<widget class="QTextEdit" name="MInputMin">
    <property name="geometry">
        <rect>
            <x>420</x>
            <y>255</y>
            <width>131</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt &quot;Tahoma&quot;;
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
    </property>
    <property name="html">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-/W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Tahoma'; font-
size:10pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
</widget>
<widget class="QLabel" name="MNameMax">
    <property name="geometry">
        <rect>
            <x>570</x>
            <y>255</y>
            <width>41</width>
            <height>41</height>
        </rect>

```



```

    </property>
    <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
        <string>Max:</string>
    </property>
    <property name="alignment">
        <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
</widget>
<widget class="QTextEdit" name="MInputMax">
    <property name="geometry">
        <rect>
            <x>620</x>
            <y>255</y>
            <width>131</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt &quot;Tahoma&quot;;
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
    </property>
    <property name="html">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot; -//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Tahoma'; font-
size:10pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
</widget>
<widget class="QLabel" name="HNameHFunc">
    <property name="geometry">
        <rect>
            <x>370</x>

```

```

        <y>50</y>
        <width>131</width>
        <height>41</height>
    </rect>
</property>
<property name="styleSheet">
    <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
</property>
<property name="text">
    <string>HFunc number:</string>
</property>
<property name="alignment">
    <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
</property>
</widget>
<widget class="QTextEdit" name="HInputFile">
    <property name="geometry">
        <rect>
            <x>710</x>
            <y>50</y>
            <width>131</width>
            <height>41</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt &quot;Tahoma&quot;;
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
    </property>
    <property name="html">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
/&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Tahoma'; font-
size:10pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
</widget>

```

```

<widget class="QPushButton" name="MPushOk">
  <property name="geometry">
    <rect>
      <x>920</x>
      <y>290</y>
      <width>50</width>
      <height>50</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">border-image: url(:/ok.png);</string>
  </property>
  <property name="text">
    <string/>
  </property>
</widget>
<widget class="QSpinBox" name="spinHFunc">
  <property name="geometry">
    <rect>
      <x>510</x>
      <y>50</y>
      <width>43</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
  </property>
  <property name="minimum">
    <number>1</number>
  </property>
  <property name="maximum">
    <number>3</number>
  </property>
</widget>
<widget class="QCustomPlot" name="outputPlot" native="true">
  <property name="geometry">
    <rect>
      <x>30</x>
      <y>440</y>
      <width>1220</width>
      <height>500</height>
    </rect>
  </property>
  <property name="styleSheet">

```

```

    <string notr="true"/>
  </property>
  <widget class="QTextEdit" name="outputText">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>1220</width>
        <height>500</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">background-color: rgb(40, 40, 40);
color: white;
font: 10pt "Tahoma";
padding: 4px 5px 0px 7px;</string>
    </property>
  </widget>
</widget>
  <widget class="QTextEdit" name="HInputOut">
    <property name="geometry">
      <rect>
        <x>710</x>
        <y>105</y>
        <width>131</width>
        <height>41</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">background-color: rgb(50, 50, 50);
color: white;
border-radius: 15px;
font: 10pt "Tahoma";
padding: 4px 5px 0px 7px;
border: 2px solid rgb(80,80,80);</string>
    </property>
  <property name="html">
    <string>&lt;!DOCTYPE HTML PUBLIC "http://www.w3.org/TR/REC-html40/strict.dtd"
&lt;html&gt;&lt;head&gt;&lt;meta name="qrichtext" content="1"
/&gt;&lt;style type="text/css&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style="font-family:'Tahoma'; font-size:10pt; font-weight:400; font-style:normal&gt;&gt;

```

```

<p style="qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;"><br /></p></body></html></string>
  </property>
</widget>
<widget class="QLabel" name="HNameOut">
  <property name="geometry">
    <rect>
      <x>600</x>
      <y>105</y>
      <width>101</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 10pt "Tahoma";
color: rgb(235, 235, 235);</string>
  </property>
  <property name="text">
    <string>Output file:</string>
  </property>
  <property name="alignment">
    <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
  </property>
</widget>
<widget class="QLabel" name="HNameLength">
  <property name="geometry">
    <rect>
      <x>370</x>
      <y>105</y>
      <width>71</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 10pt "Tahoma";
color: rgb(235, 235, 235);</string>
  </property>
  <property name="text">
    <string>Factor:</string>
  </property>
  <property name="alignment">
    <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
  </property>
</widget>
<widget class="QSpinBox" name="spinFactor">

```

```

    <property name="geometry">
      <rect>
        <x>452</x>
        <y>105</y>
        <width>101</width>
        <height>41</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">font: 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="minimum">
      <number>1</number>
    </property>
    <property name="maximum">
      <number>1000</number>
    </property>
    <property name="value">
      <number>1</number>
    </property>
  </widget>
  <widget class="QRadioButton" name="MTestPairs">
    <property name="geometry">
      <rect>
        <x>440</x>
        <y>220</y>
        <width>141</width>
        <height>22</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
      <string>Pairs num</string>
    </property>
  </widget>
  <widget class="QRadioButton" name="MTestKey">
    <property name="geometry">
      <rect>
        <x>600</x>
        <y>220</y>
        <width>131</width>
        <height>22</height>

```

```

    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
  </property>
  <property name="text">
    <string>Key length</string>
  </property>
</widget>
<widget class="QRadioButton" name="MTestFactor">
  <property name="geometry">
    <rect>
      <x>760</x>
      <y>220</y>
      <width>131</width>
      <height>22</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
  </property>
  <property name="text">
    <string>Factor</string>
  </property>
</widget>
<widget class="QSpinBox" name="spinStep">
  <property name="geometry">
    <rect>
      <x>842</x>
      <y>255</y>
      <width>61</width>
      <height>41</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">font: 11pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
  </property>
  <property name="minimum">
    <number>1</number>
  </property>
  <property name="maximum">
    <number>1000</number>
  </property>

```

```

    <property name="value">
      <number>1</number>
    </property>
  </widget>
  <widget class="QLabel" name="HNameStep">
    <property name="geometry">
      <rect>
        <x>770</x>
        <y>255</y>
        <width>51</width>
        <height>41</height>
      </rect>
    </property>
    <property name="styleSheet">
      <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
    </property>
    <property name="text">
      <string>Step:</string>
    </property>
    <property name="alignment">
      <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
    </property>
  </widget>
  <widget class="QWidget" name="horizontalLayoutWidget">
    <property name="geometry">
      <rect>
        <x>480</x>
        <y>310</y>
        <width>361</width>
        <height>31</height>
      </rect>
    </property>
    <layout class="QHBoxLayout" name="MLayout">
      <item>
        <widget class="QRadioButton" name="MTestSum">
          <property name="styleSheet">
            <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
          </property>
          <property name="text">
            <string>Iterations sum</string>
          </property>
        </widget>
      </item>
    </layout>
  </widget>

```



```

    <widget class="QRadioButton" name="MTestMax">
      <property name="layoutDirection">
        <enum>Qt::RightToLeft</enum>
      </property>
      <property name="styleSheet">
        <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
      </property>
      <property name="text">
        <string>Longest insert</string>
      </property>
    </widget>
  </item>
</layout>
</widget>
<widget class="Line" name="OutLine">
  <property name="geometry">
    <rect>
      <x>30</x>
      <y>360</y>
      <width>1220</width>
      <height>3</height>
    </rect>
  </property>
  <property name="styleSheet">
    <string notr="true">background-color: rgb(70, 70, 70);</string>
  </property>
  <property name="orientation">
    <enum>Qt::Horizontal</enum>
  </property>
</widget>
<widget class="QWidget" name="horizontalLayoutWidget_2">
  <property name="geometry">
    <rect>
      <x>30</x>
      <y>390</y>
      <width>281</width>
      <height>31</height>
    </rect>
  </property>
  <layout class="QHBoxLayout" name="OutLayout">
    <item>
      <widget class="QRadioButton" name="OutSetPlot">
        <property name="styleSheet">
          <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>

```

```

    </property>
    <property name="text">
        <string>Plot output</string>
    </property>
</widget>
</item>
<item>
    <widget class="QRadioButton" name="OutSetText">
        <property name="layoutDirection">
            <enum>Qt::RightToLeft</enum>
        </property>
        <property name="styleSheet">
            <string notr="true">font: 10pt &quot;Tahoma&quot;;
color: rgb(235, 235, 235);</string>
        </property>
        <property name="text">
            <string>Text output</string>
        </property>
    </widget>
</item>
</layout>
</widget>
<widget class="QPushButton" name="OutSaveFile">
    <property name="geometry">
        <rect>
            <x>1200</x>
            <y>380</y>
            <width>50</width>
            <height>50</height>
        </rect>
    </property>
    <property name="styleSheet">
        <string notr="true">border-image: url(:/savef.png);</string>
    </property>
    <property name="text">
        <string/>
    </property>
</widget>
<zorder>GPushOk</zorder>
<zorder>GInputPairs</zorder>
<zorder>GNameMain</zorder>
<zorder>HNameMain</zorder>
<zorder>HPushOk</zorder>
<zorder>GInputLength</zorder>
<zorder>GNamePairs</zorder>
<zorder>GNameLength</zorder>

```

```

<zorder>GNameFile</zorder>
<zorder>GInputFile</zorder>
<zorder>HNameFile</zorder>
<zorder>VLine</zorder>
<zorder>HLine</zorder>
<zorder>MNameMain</zorder>
<zorder>GCheckWorst</zorder>
<zorder>MNameMin</zorder>
<zorder>MInputMin</zorder>
<zorder>MNameMax</zorder>
<zorder>MInputMax</zorder>
<zorder>HNameHFunc</zorder>
<zorder>HInputFile</zorder>
<zorder>MPushOk</zorder>
<zorder>spinHFunc</zorder>
<zorder>HInputOut</zorder>
<zorder>HNameOut</zorder>
<zorder>HNameLength</zorder>
<zorder>spinFactor</zorder>
<zorder>MTestPairs</zorder>
<zorder>MTestKey</zorder>
<zorder>MTestFactor</zorder>
<zorder>spinStep</zorder>
<zorder>HNameStep</zorder>
<zorder>outputPlot</zorder>
<zorder>horizontalLayoutWidget</zorder>
<zorder>OutLine</zorder>
<zorder>horizontalLayoutWidget_2</zorder>
<zorder>OutSaveFile</zorder>
</widget>
<widget class="QMenuBar" name="menubar">
  <property name="geometry">
    <rect>
      <x>0</x>
      <y>0</y>
      <width>1280</width>
      <height>25</height>
    </rect>
  </property>
</widget>
<widget class="QStatusBar" name="statusbar"/>
</widget>
<customwidgets>
  <customwidget>
    <class>QCustomPlot</class>
    <extends>QWidget</extends>
  </customwidget>
</customwidgets>

```

```
    <header>qcustomplot.h</header>
    <container>1</container>
</customwidget>
</customwidgets>
<resources/>
<connections/>
</ui>
```