Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
Пермский национальный исследовательский политехнический университет
Электротехнический факультет
Кафедра информационных технологий и автоматизированных систем

Лабораторная работа №11

"Стэки"

Вариант: 12

Выполнил студент ИВТ-24-2б:
Шишкин Максим Григорьевич

_____
(дата, подпись)

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

_____
(дата, подпись)

Пермь 2025

Содержание

1 Постановка задачи

Записи в стэке содержат ключевое поле типа *char(строка символов). Сформировать стэк. Удалить из него Элементы, с одинаковыми ключевыми полями. Добавить элемент после элемента с заданным ключевым полем.

2 Код на C++

```cpp
#include <iostream>
#include <string>
#include <clocale>

using namespace std;

const int MAX_SIZE = 100; // Максимальный размер стека

struct Stack {
    string items[MAX_SIZE];
    int top;  // Индекс верхнего элемента
};

// Инициализация стека
void initialize(Stack& s) {
    s.top = -1; // Стек пуст
}

// Проверка на пустоту стека
bool isEmpty(const Stack& s) {
    return s.top == -1;
}

// Проверка на заполненность стека
bool isFull(const Stack& s) {
    return s.top == MAX_SIZE - 1;
}

// Добавление элемента в стек (push)
bool push(Stack& s, const string& key) {
    if (isFull(s)) {
        cout << "Стек переполнен!" << endl;
        return false;
    }
    s.top++;
    s.items[s.top] = key;
    return true;
}

// Удаление элемента из стека (pop)
string pop(Stack& s) {
    if (isEmpty(s)) {
        cout << "Стек пуст!" << endl;
        return ""; // Или другое значение по умолчанию
    }
    string temp = s.items[s.top];
    s.top--;
```

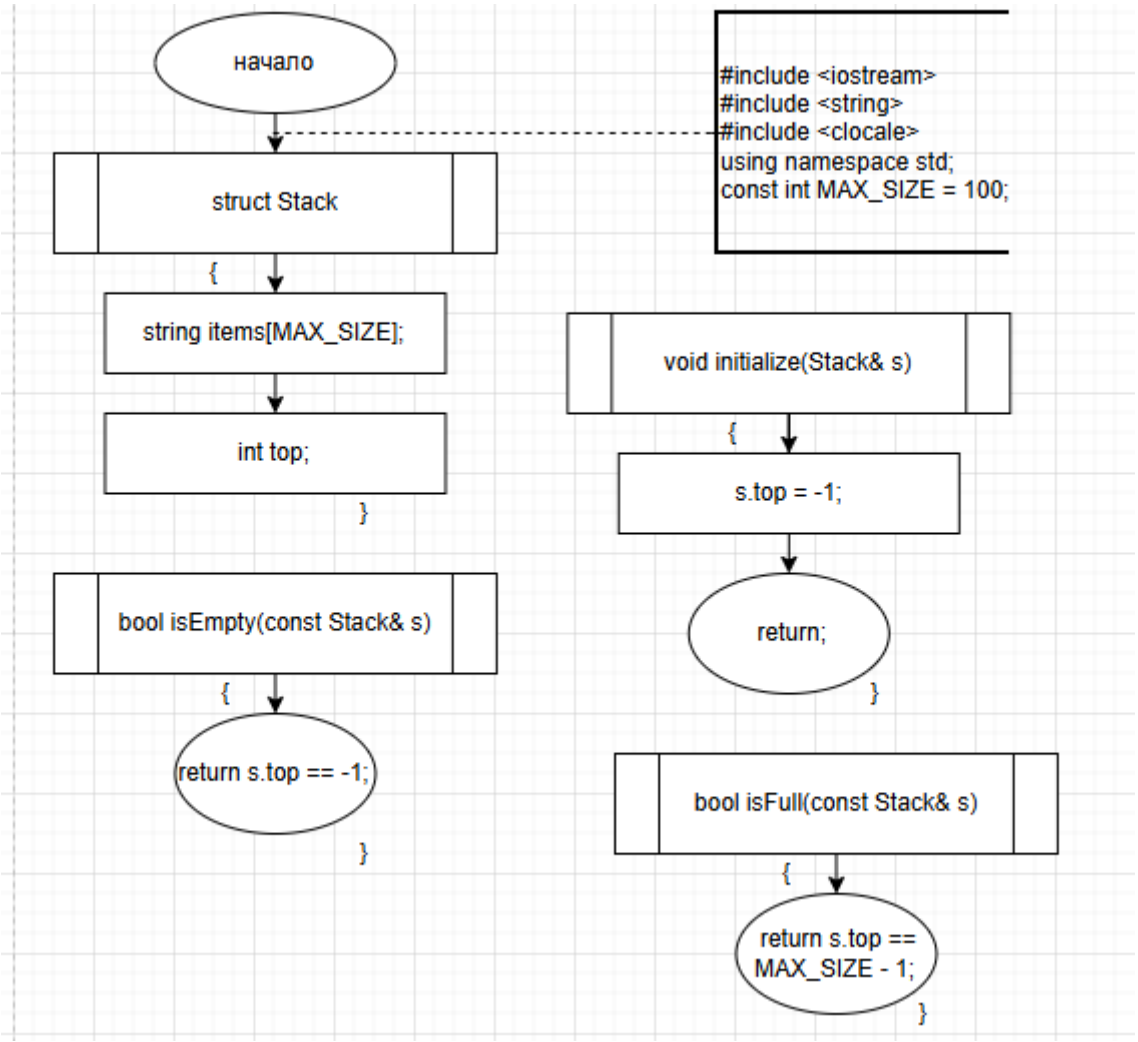
```
48      return temp;
49  }
50
51  // Печать стека (без изменения его содержимого)
52  void printStack(const Stack& s) {
53      if (isEmpty(s)) {
54          cout << "Стек пуст!" << endl;
55          return;
56      }
57      cout << "Стек: ";
58      for (int i = 0; i <= s.top; ++i) {
59          cout << s.items[i] << " ";
60      }
61      cout << endl;
62  }
63
64  // Добавление элемента после заданного (для стека не имеет особого смысла, но можно реализовать)
65  bool addAfter(Stack& s, const string& targetKey, const string& newKey) {
66      Stack tempStack;
67      initialize(tempStack);
68      bool found = false;
69
70      // Перекладываем элементы из исходного стека во временный
71      while (!isEmpty(s)) {
72          string item = pop(s);
73          if (item == targetKey && !found) {
74              push(tempStack, item);
75              push(tempStack, newKey);
76              found = true;
77          }
78          else {
79              push(tempStack, item);
80          }
81      }
82
83      // Возвращаем элементы обратно в исходный стек (в обратном порядке)
84      while (!isEmpty(tempStack)) {
85          push(s, pop(tempStack));
86      }
87
88      if (!found) {
89          cout << "Элемент с ключом '" << targetKey << "' не найден." << endl;
90      }
91      return found;
92  }
93
94  // Удаление дубликатов (довольно сложная операция для стека)
```

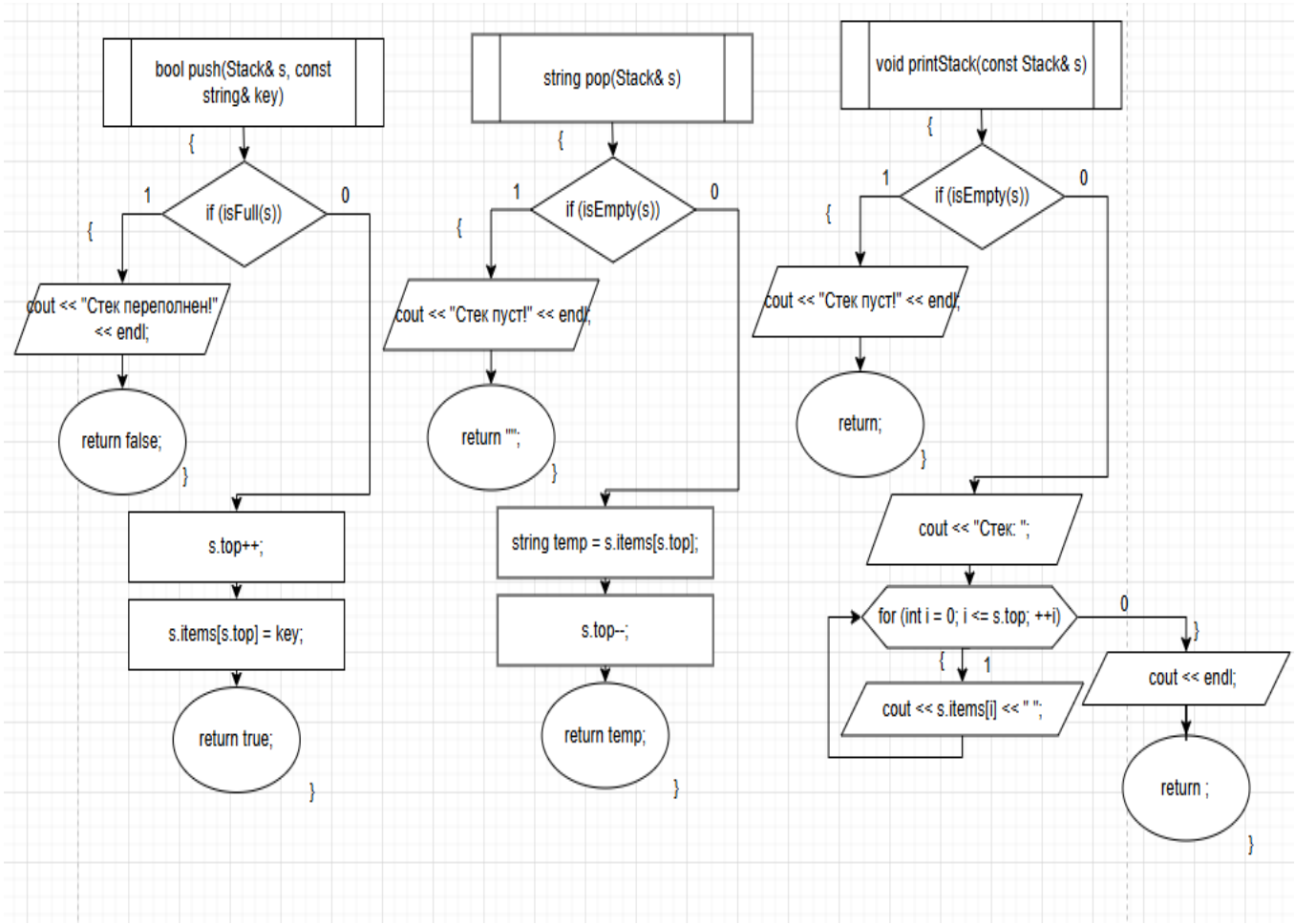
```cpp
void removeDuplicates(Stack& s) {
    Stack tempStack;
    initialize(tempStack);
    string uniqueItems[MAX_SIZE];
    int uniqueCount = 0;

    while (!isEmpty(s)) {
        string item = pop(s);
        bool duplicate = false;
        for (int i = 0; i < uniqueCount; ++i) {
            if (uniqueItems[i] == item) {
                duplicate = true;
            }
        }
        if (!duplicate) {
            uniqueItems[uniqueCount] = item;
            uniqueCount++;
            push(tempStack, item);
        }
    }

    // Возвращаем уникальные элементы обратно в исходный стек (в обратном порядке)
    for (int i = uniqueCount - 1; i >= 0; --i) {
        push(s, uniqueItems[i]);
    }
}

int main() {
    setlocale(LC_ALL, "ru");
    string strArr[] = { "sos", "acer", "dog", "sos", "date", "acer", "cat" };
    int sz = sizeof(strArr) / sizeof(strArr[0]);

    Stack myStack;
    initialize(myStack);

    cout << "Исходный массив: ";
    for (int i = 0; i < sz; ++i) {
        cout << strArr[i] << " ";
    }
    cout << endl;

    // Заполняем стек из массива (в обратном порядке, чтобы первый элемент массива был на вершине стека)
    for (int i = sz - 1; i >= 0; --i) {
        push(myStack, strArr[i]);
    }

    cout << "Стек: ";
    printStack(myStack);

    removeDuplicates(myStack);
    cout << "Стек после удаления дубликатов: ";
    printStack(myStack);

    addAfter(myStack, "dog", "mouse");
    cout << "Стек после добавления элемента: ";
    printStack(myStack);

    return 0;
}
```

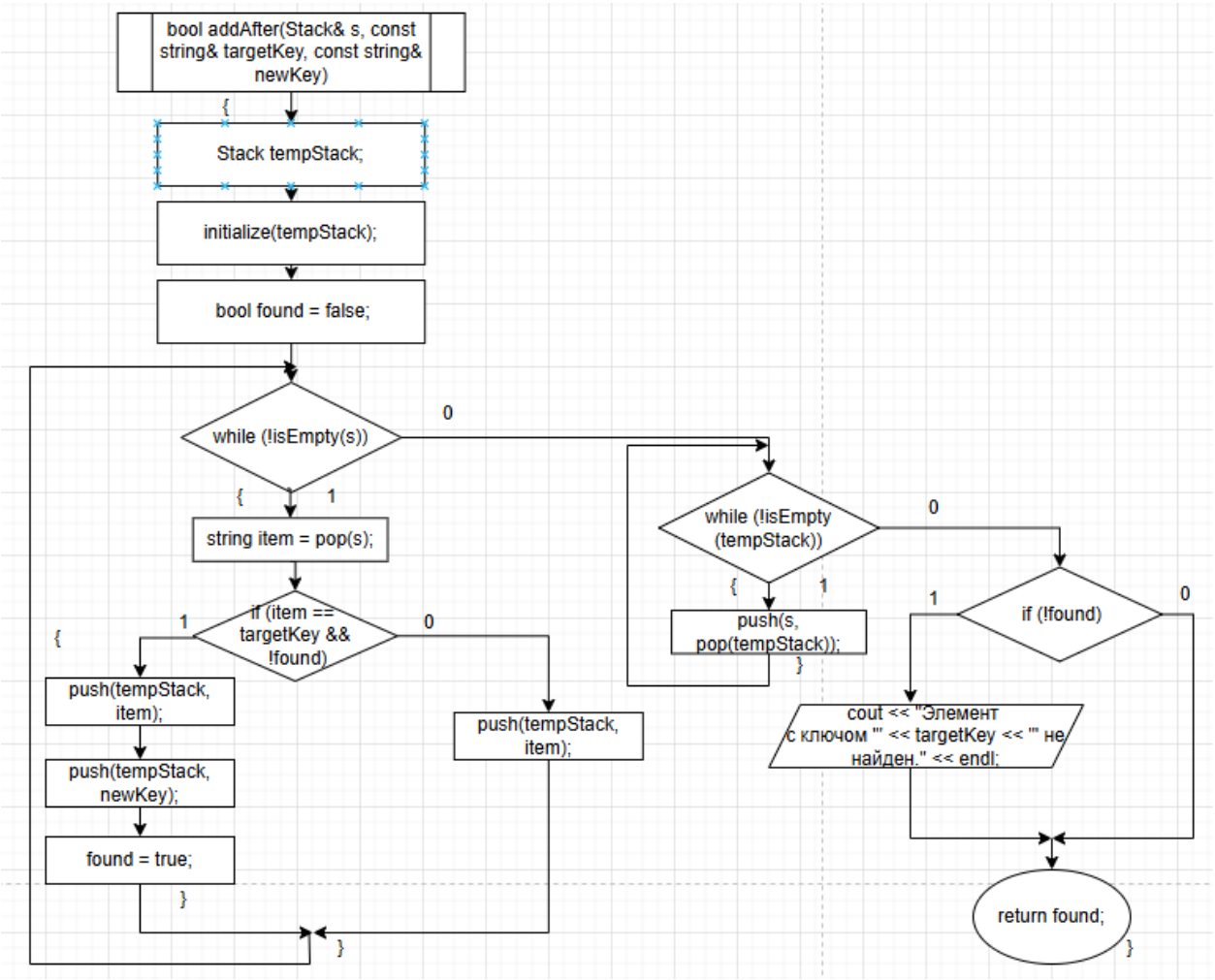
3 Результат работы программы

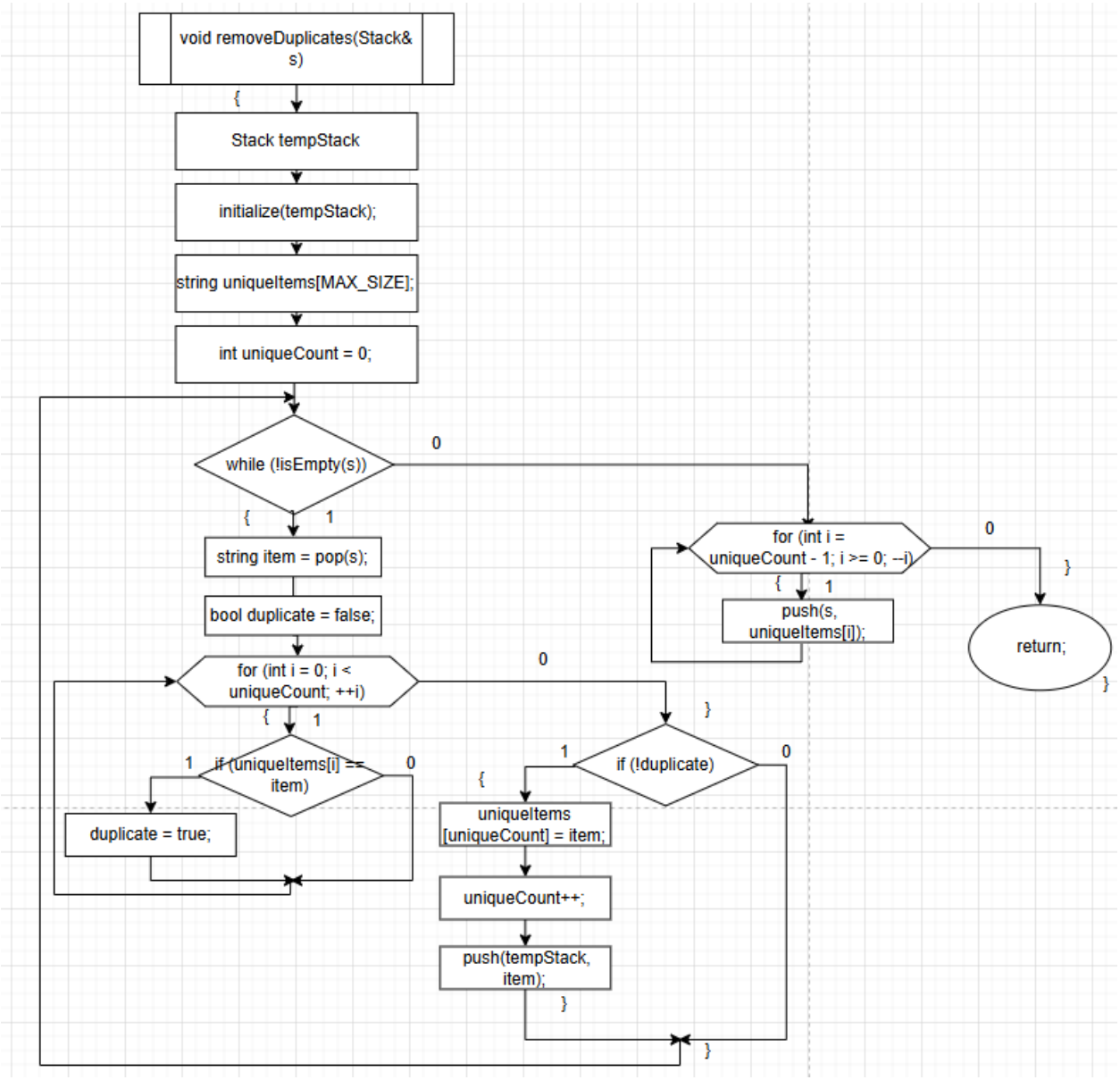
```
Исходный массив: sos acer dog sos date acer cat
Стек: Стек: cat acer date sos dog acer sos
Стек после удаления дубликатов: Стек: cat date dog acer sos
Стек после добавления элемента: Стек: cat date mouse dog acer sos
```

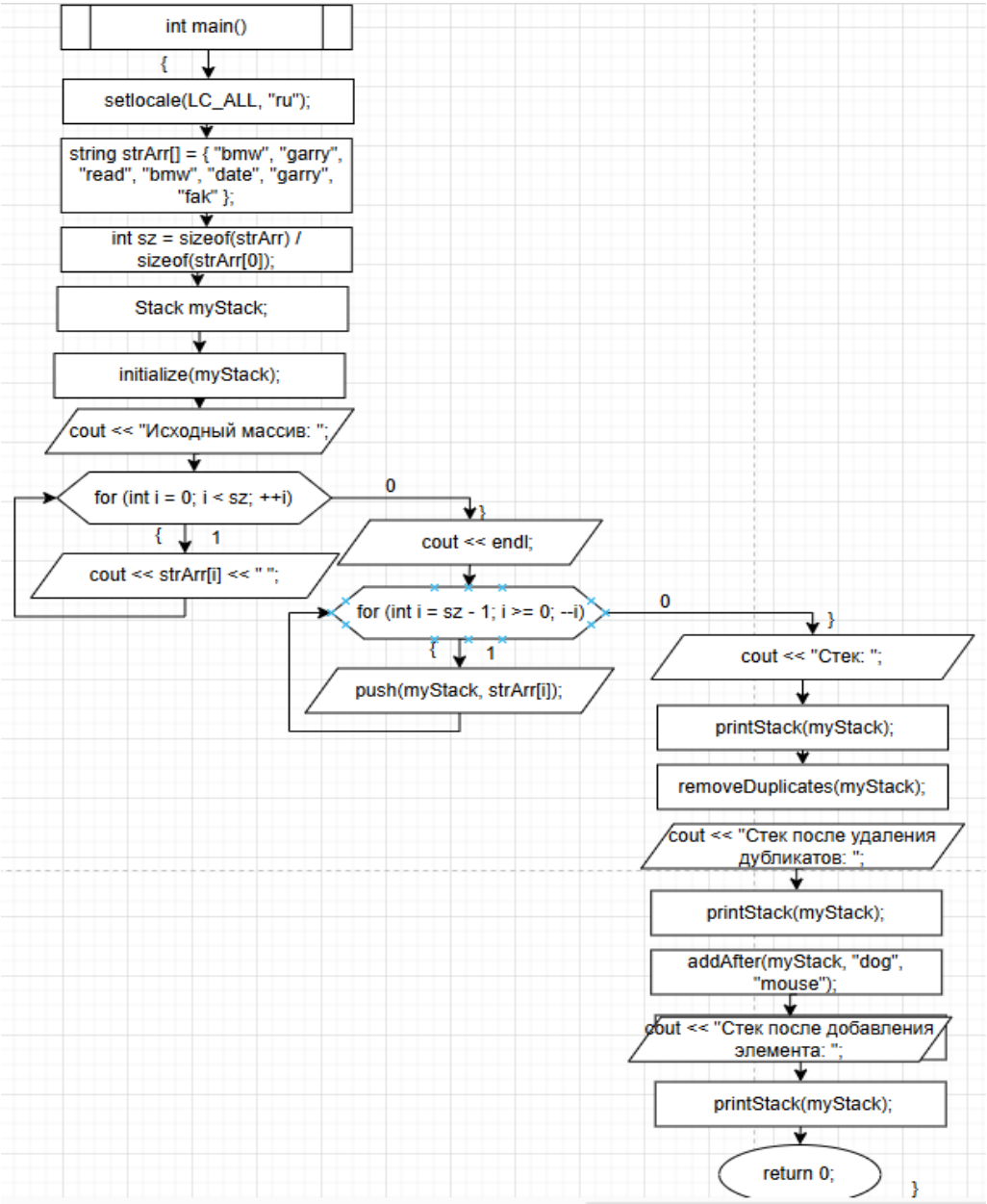
4 Блок-схема




bool push(Stack& s, const string& key)
{
if (isFull(s))
1
0
{
cout << "Стек переполнен!" << endl;
return false;
}
s.top++;
s.items[s.top] = key;
return true;
}
string pop(Stack& s)
{
if (isEmpty(s))
1
0
{
cout << "Стек пуст!" << endl;
return "";
}
string temp = s.items[s.top];
s.top--;
return temp;
}
void printStack(const Stack& s)
{
if (isEmpty(s))
1
0
{
cout << "Стек пуст!" << endl;
return;
}
cout << "Стек: ";
for (int i = 0; i <= s.top; ++i)
0
{
1
cout << s.items[i] << " ";
}
cout << endl;
return ;
}


bool addAfter(Stack& s, const string& targetKey, const string& newKey)
{
Stack tempStack;
initialize(tempStack);
bool found = false;
while (!isEmpty(s))
0
{
1
string item = pop(s);
if (item == targetKey && !found)
1
0
{
push(tempStack, item);
push(tempStack, newKey);
found = true;
}
push(tempStack, item);
}
while (!isEmpty (tempStack))
0
{
1
push(s, pop(tempStack));
}
if (!found)
1
0
cout << "Элемент с ключом '" << targetKey << "' не найден." << endl;
return found;
}


void removeDuplicates(Stack&
s)
{
Stack tempStack
initialize(tempStack);
string uniqueItems[MAX_SIZE];
int uniqueCount = 0;
while (!isEmpty(s))
0
{
1
1
string item = pop(s);
bool duplicate = false;
for (int i = 0; i <
uniqueCount; ++i)
0
{
1
1
if (uniqueItems[i] ==
item)
0
duplicate = true;
}
1
if (!duplicate)
0
{
uniqueItems
[uniqueCount] = item;
uniqueCount++;
push(tempStack,
item);
}
}
for (int i =
uniqueCount - 1; i >= 0; --i)
0
{
1
push(s,
uniqueItems[i]);
}
return;
}



5 Ссылка на github

ссылка на github - https://github.com/MAKSPOWERO/mas1