ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| Обработка текстовых данных |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № |  |  |  |  | Ташев Д.Е. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

**1.Цель работы:** Целью работы является изучение представления строковых данных в языке С и алгоритмов их обработки.

**2.Задачи работы:** Разработать функцию, которая выполняет ту обработку символьной строки, которая определена в Вашем индивидуальном задании.

Функция находит в строке первый символ, который не входит в другую заданную строку

**3.Описание функций:**  
main-основная функция  
**1)Имя: main**  
**Назначение:** ввод пользователем первой и второй строки и поиск символа, который которые не вошли во вторую строку из первой.

**Входные данные:** нет  
**Выходные данные:** нет  
**Побочный эффект:** отсутствует.

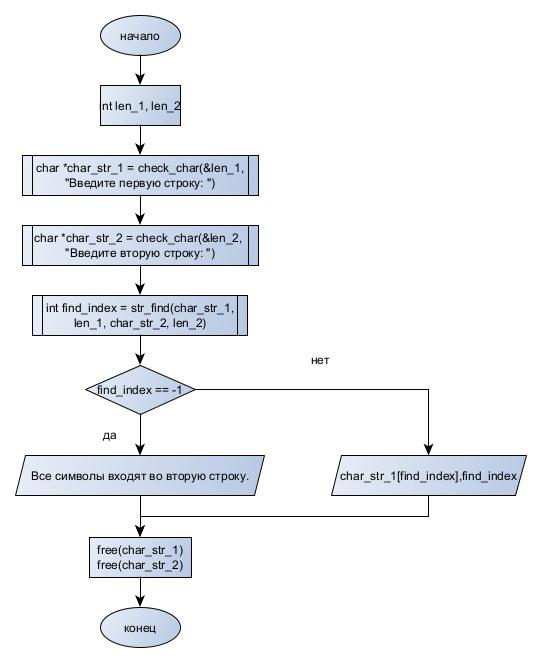
**Тестовые данные:**   
**Ввод: hello world**

**guap world**

**Вывод:** **Первый символ, который мы нашли: l**  
**Прототип:** int main()  
**Псевдокод:**ввод первой строки пользователем

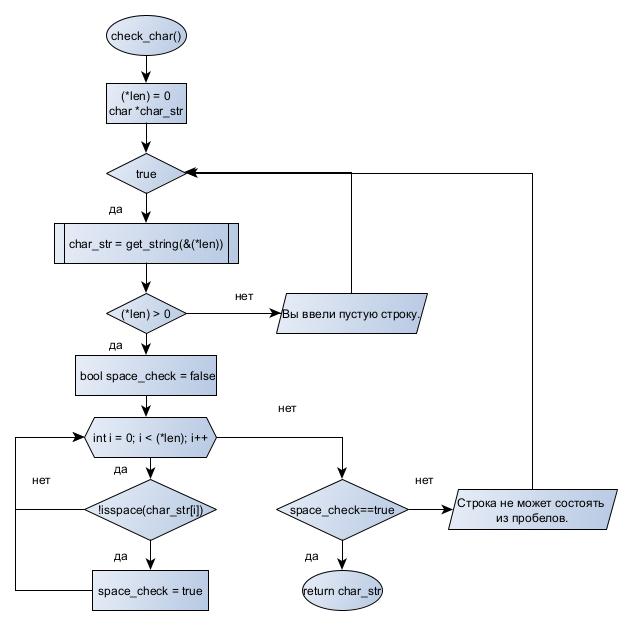
ввод второй строки пользователем

поиск символа, который входит во вторую строку из первой

если функция такого символа не нашла, то вывод сообщения, что символы входят  
**Блок-схема:**  
  
  
**2)Имя: check\_char()**  
**Назначение:** функция для ввода строки и проверки её на заполнение  
**Входные данные:** нет  
**Выходные данные:** char\_str  
**Побочный эффект:** отсутствует.   
**Прототип:** char\* check\_char()  
**Псевдокод:**Функция ввода строки

Если символов больше 0, то проверка на пробелы

Если количество пробелов=количество символов, то ввод строки заново

Если символов 0, то ввод строки заново  
**Блок-схема:**  
****

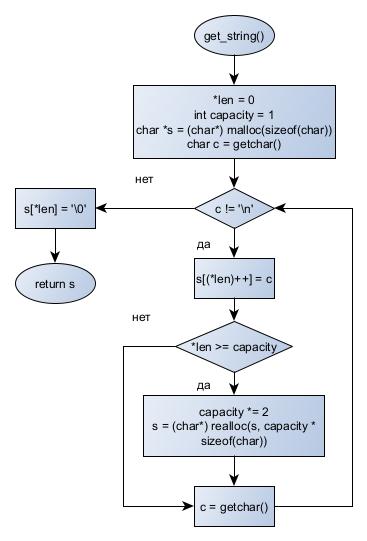
**3)Имя: get\_string()**  
**Назначение:** функция для ввода строки  
**Входные данные:** нет  
**Выходные данные:** s  
**Побочный эффект:** отсутствует.   
**Прототип:** char\* get\_string()  
**Псевдокод:**Создание массива

Получение символа

Если этот символ не равен ‘\n’, то запись в массив

Если равен ‘\n’, то конец функции

**Блок-схема:**

  
**4)Имя: str\_find()**  
**Назначение:** функция нахождения в строке символа, который входит в другую строку  
**Входные данные:** \*str\_1, \*str\_2, len\_1, len\_2  
**Выходные данные:** i  
**Побочный эффект:** отсутствует.   
**Прототип:** int str\_find()  
**Псевдокод:**Создание флага состояния поиска

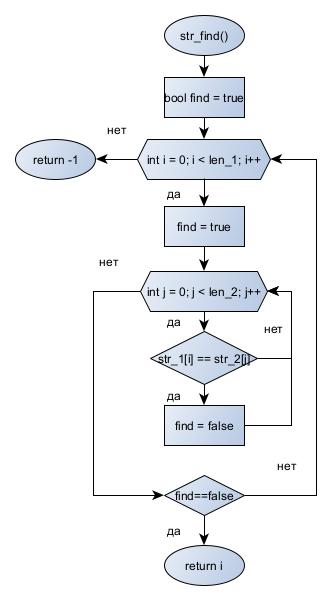
Перебирается первая строка + обновление флага на состояние true

В процессе перебора перебирается вторая строка, если символы совпадают, то флан=false

Если находится символ, который мы не нашли, то возвращаем его

Если все символы нашли, то возвращаем -1

**Блок-схема:**

****

**5.ЛИСТИНГ КОДА:**

int str\_find(char \*str\_1, int len\_1, char \*str\_2, int len\_2) {

// флаг состояния поиска

bool find = true;

// для 1 строки

for (int i = 0; i < len\_1; i++) {

find = true;

// цикл для 2 строки

for (int j = 0; j < len\_2; j++)

// если нашли одинаковые символы, то идём на следующую итерацию цикла

if (str\_1[i] == str\_2[j]) {

find = false;

break;

}

// если есть символ который мы не нашли, то возвращаем его

if (!find) return i;

}

// если строки одинаковые (по набору символов), то возвращаем -1

return -1;

}

// функция для ввода строки и проверки её

char \*check\_char(int \*len, const char \*promt = "") {

(\*len) = 0;

char \*char\_str;

// ввод строки

while (true) {

cout << promt;

char\_str = get\_string(&(\*len));

if ((\*len) > 0) {

// проверка на пробелы (строка не должна состоять из пробелов)

bool space\_check = false;

for (int i = 0; i < (\*len); i++)

if (!isspace(char\_str[i])) {

space\_check = true;

break;

}

if (space\_check)

break;

else

cout << "Строка не может состоять из пробелов." << endl;

} else {

cout << "Вы ввели пустую строку." << endl;

}

}

return char\_str;

}

int main() {

// смена кодировки

system("chcp 65001");

// очистка терминала

//clear\_scr();

// здесь будет храниться длинна вводимых строк

int len\_1, len\_2;

// вводим строки

char \*char\_str\_1 = check\_char(&len\_1, "Введите первую строку: ");

char \*char\_str\_2 = check\_char(&len\_2, "Введите вторую строку: ");

draw\_line(20);

//

int find\_index = str\_find(char\_str\_1, len\_1, char\_str\_2, len\_2);

if (find\_index == -1) {

cout << "Все символы входят во вторую строку." << endl;

} else {

cout << "Первый символ который мы нашли: " << char\_str\_1[find\_index] << endl;

cout << "Его индекс в первой строке: " << find\_index << endl;

}

// очищаем память

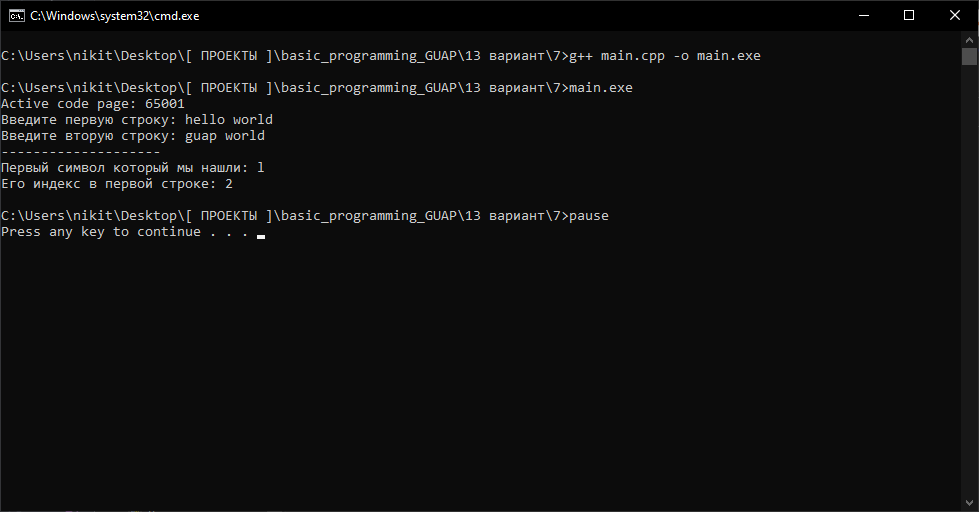
free(char\_str\_1);

free(char\_str\_2);

return 0;

}

**6.Пример выполнения программы:**



**7.Анализ результатов и выводы:**

В ходе этой лабораторной работы мы изучили возможность работы с одномерными массивами.

Из плюсов можно отметить, что поиск символа происходит с учетом регистра.

Существенных минусов в программе не обнаружено.