ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

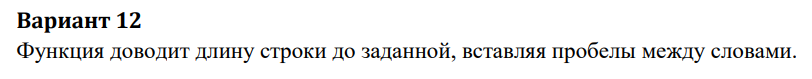
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

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

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4135К |  |  |  | Столяров Н.С. |
|  |  |  | 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:**  
  
**3.Описание функций:  
Имя: main**   
**Назначение:** Вычисление номера первой строки содержащей положительный элемент. Удаление столбцов и строк заполненными нулями.  
**Входные данные:** Высота, ширина и массив.   
**Выходные данные:** Индекс и массив.  
**Побочный эффект:** отсутствует.   
**Тестовые данные:**

|  |  |  |
| --- | --- | --- |
| Входная строка | Длинна новой строки | Ответ |
| hello world | 20 | hello world Длинна новой строки: 20 |

**Прототип:** int main()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| Ввод строки  Ввод длинны строки которую хотим получить  Увеличиваем строку  Выводим новую строку |  |

**Имя: add\_space\_char**  
**Назначение:** Увеличение строки (вставляет пробелы между словами)  
**Входные данные:** char\_str, len, len\_new.   
**Выходные данные:** char\_str.  
**Побочный эффект:** отсутствует.   
**Прототип:** char \*add\_space\_char()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| * Пока длинна введённой строки меньше чем ту которую мы хотим получить * Если в строке встречается пробел или это первый символ * Сдвигаем строку направо * Вставляем пробел * Увеличиваем длину строки на 1 |  |

**Имя: read\_value**  
**Назначение:** проверка переменной a на корректность ввода  
**Входные данные:** Отсутствуют.   
**Выходные данные:** x(double).  
**Побочный эффект:** отсутствует.   
**Прототип:** double read\_value()

|  |  |
| --- | --- |
| **Псевдокод** | **Блок-схема** |
| Бесконечный цикл  ввод строки  проверка на пробелы и удаление их  проверка на запятые и замена их на точки  проверка на минус  проверка на лишние символы  если не было проблем, то останавливаем бесконечный цикл­  конец бесконечного цикла  перевод строки в число  вывод числа |  |

**5.ЛИСТИНГ КОДА:**

**main.cpp  
/\*   
 Разработать функцию, которая выполняет ту обработку символьной строки, которая   
 определена в Вашем индивидуальном задании   
  
 Вариант 12   
   Функция доводит длину строки до заданной, вставляя пробелы между словами.   
  
\*/   
  
#include <iostream>   
using namespace std;   
  
#include "libs/lib.h"   
#include <cmath>   
#include <time.h>   
  
// проверка ввода   
#include "libs/simple\_char.h"   
#include "libs/input\_validation.h"   
  
#include "more\_char.h"   
  
int main() {   
       // смена кодировки   
 system("chcp 65001");   
  
 // очистка терминала   
 //clear\_scr();   
  
 int len;   
 char \*char\_str;   
  
 // ввод строки   
 while (true) {   
   cout << "Введите строку: ";   
   char\_str = get\_string(&len);   
  
   if (len > 0) {   
     break;   
   } else {   
     cout << "Вы ввели пустую строку." << endl;   
   }   
 }   
  
 // вывод длинны текущей строки   
 draw\_line(20);   
 cout << "Длинна введённой строки: " << len << endl;   
 draw\_line(20);   
  
 // ввод длинны новой строки   
 int len\_new;   
 while (true) {   
   len\_new = read\_value("Введите длинну новой строки: ", false, false, false);   
  
   if (len\_new < len) {   
     cout << "Длинна новой строки должна быть больше старой строки (" << len << ")." << endl;   
   } else break;   
 }   
  
 draw\_line(20);   
  
 char\_str = add\_space\_char(char\_str, &len, len\_new);   
  
 // выводим новую строку   
 cout << char\_str << endl;   
 draw\_line(20);   
 cout << "Длинна новой строки: " << len << endl;   
  
 free(char\_str);   
  
       return 0;   
}  
  
more\_char.h**

#include <iostream>

using namespace std;

char \*add\_space\_char(char \*char\_str, int \*len, int len\_new) {

int j = 1;

int i = 0;

// основной код

while ((\*len) < len\_new) {

// если встречаем пробел или мы на первом символе

if (char\_str[i] == ' ' || i == 0) {

(\*len)++;

char\_str = (char\*) realloc(char\_str, (\*len+1) \* sizeof(char));

for (int k = (\*len); k > i; k--) {

char\_str[k] = char\_str[k-1];

}

char\_str[i] = ' ';

i += j;

}

// счётсчики

i++;

if (i >= (\*len)) {

i = 0;

j++;

}

}

return char\_str;

}

**simple\_char.h**

#include <iostream>

char \*get\_string(int \*len) {

\*len = 0;

int capacity = 1;

char \*s = (char\*) malloc(sizeof(char));

char c = getchar();

while (c != '\n') {

s[(\*len)++] = c;

if (\*len >= capacity) {

capacity \*= 2;

s = (char\*) realloc(s, capacity \* sizeof(char));

}

c = getchar();

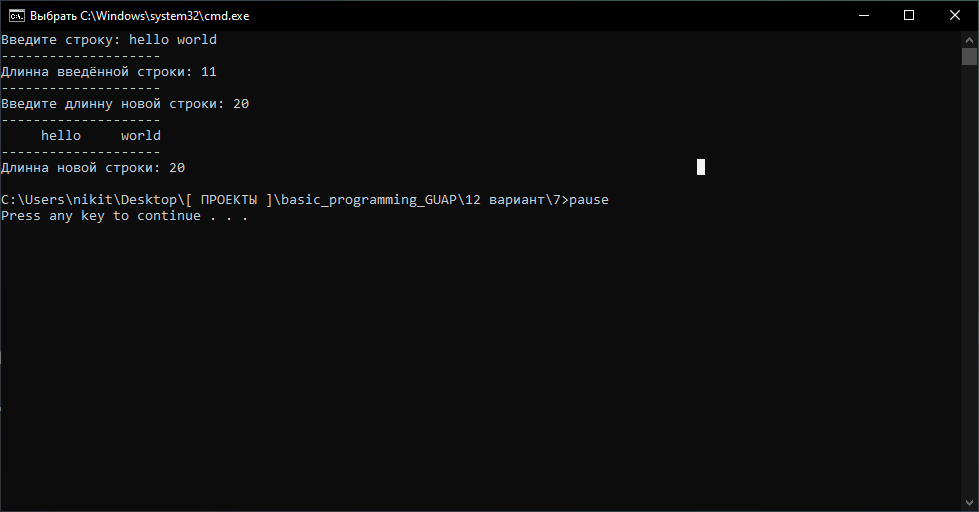
}

s[\*len] = '\0';

return s;

}

**10.Пример выполнения программы:**



Видно, что результаты расчётов совпадают с тестовыми данными.

**11.Анализ результатов и выводы:**

В ходе этой лабораторной работы мы научились работать со строками (массивами char).  
Из плюсов можно отметить её работоспособность и простоту.

Из минусов ничего, всё работает как часы.