Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Ордена Трудового Красного Знамени

федеральное государственное бюджетное образовательное учреждение

высшего образования

Московский технический университет связи и информатики

(МТУСИ)

Кафедра «Информационная безопасность»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

по дисциплине

«Программирование в системах информационной безопасности»

на тему

«Строки»

Вариант №19

Выполнил:

студент группы БСУ1801 Плотников П.С

Проверил:

старший преподаватель кафедры ИБ

Барков В. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2021

**Цель**

Овладеть навыками работы со строками в языке программирования C.

# **Задание 1**

Написать функцию подсчёта дины строки. Функция принимает исходную строку и возвращает количество символов в строке, не включая нулевой символ.

Прототип функции int strlen(const char \*str).

Функция не должна использовать функции консольного ввода-вывода.

Листинг №1 - Исходный код файла task1.c

unsigned int strlen(const char \*str)

{

int len = 0;

while(\*str++) ++len;

return len;

}

# **Задание 2**

Написать функцию, поиска подстроки в строке. Функция принимает исходную строку и подстроку для поиска и возвращает адрес подстроки, если подстрока найдена, 0 в противном случае. Обращение к символам строки после нулевого символа недопустимо.

Прототип функции char \*find(const char \*str, const char \*substr).

Функция не должна использовать функции консольного ввода-вывода.

Листинг 2 – Исходный код файла task2.c

#include "func.h"

char \*find(const char\* str, const char\* substr)

{

unsigned int i = 0;

unsigned int j,k;

int sbstrlen = strlen1(substr);

while (str[i] != 0)

{

for (j = i, k = 0; substr[k] != 0 && str[j] == substr[k]; j++,k++)

{

if (substr[k+1] == 0)

{

return (unsigned char \*)str + i;

}

}

++i;

}

return 0;

}

# **Задание 3**

Написать функцию, удаления подстроки в строке. Функция принимает исходную строку и подстроку для поиска. С помощью функции find найти адрес начала подстроки и затем удалить эту подстроку. Оставшиеся в конце символы заполнить нулями. Обращение к символам строки после нулевого символа недопустимо.

Прототип функции void delete(char \*str, const char \*substr).

Функция не должна использовать функции консольного ввода-вывода.

Листинг 3 – Исходный код файла task3.c

#include "func.h"

void delete(char \*str, const char \*substr)

{

int substrlen = strlen(substr);

char \*point = find(str,substr);

for (int i = 0; \*(point + substrlen + i) != 0 ; i++)

{

\*(point + i) = \*(point + substrlen + i);

\*(point + substrlen + i) = 0;

}

}

# **Задание 4**

Написать функцию копирования строк. Функция принимает исходную строку, буфер для копирования и размер буфера. Если длина исходной строки больше размера буфера, необходимо скопировать столько символов, сколько вместится в буфер, не забыв добавить в конце строки нулевой символ.

Прототип функции void strcpy(char \*dest, int len, const char \*src).

Функция не должна использовать функции консольного ввода-вывода.

Листинг 4 – Исходный код файла task4.c

#include <memory.h>

void strcpy(char \*dest, int len, const char\* src )

{

if (len <=0)

return;

memset(dest, 0, len);

char\* curDst = dest;

for(const char\* curSrc = src; \*curSrc && curDst - dest <= len - 1; ++curSrc, ++curDst)

{

\*curDst = \*curSrc;

}

}

# **Задание 5**

Написать функцию конкатенации строк. Функция принимает исходные строки, буфер для размещения и размер буфера. Если сумма длин исходных строк больше размера буфера, необходимо скопировать столько символов, сколько вместится в буфер, не забыв добавить в конце строки нулевой символ.

Прототип функции void strcat(char \*dest, int len, const char \*first, const char \*second).

Функция не должна использовать функции консольного ввода-вывода.

Листинг 5 – Исходный код файла task5.c

#include "func.h"

void strcat(char \*dest, int len, const char \*first, const char \*second)

{

strcpy(dest, len, first);

unsigned firstLen = strlen(first);

strcpy(dest + firstLen, len - firstLen, second);

}

# **Задание 6**

Написать функцию вставки подстроки в строку. Функция принимает исходную строку, строку для вставки, индекс, определяющий место вставки, буфер для размещения новой строки и размер буфера. Если сумма длин исходной строки и строки для вставки больше размера буфера или индекс находится за пределами строки, необходимо вернуть соответствующую ошибку.

Прототип функции \_Bool insert(const char \*src, const char \*str, int index, char \*dest, int len).

Функция не должна использовать функции консольного ввода-вывода.

Листинг 6– Исходный код файла task6.c

#include "func.h"

\_Bool insert(const char \*src, const char \*str,int index,char \*dest,int len)

{

unsigned int strLenght = strlen(str);

unsigned int srcLen = strlen(src);

if ((strLenght + srcLen > len) || (index > strLenght)) {return 0;}

char \*point = dest;

strcpy(point,srcLen - index,src);

point += index;

strcpy(point,strLenght,str);

point += strLenght;

strcpy(point,len,src + index);

}

# **Задание 7**

Написать функцию main, демонстрирующую работу ранее разработанных функций

Листинг 7 – Исходный код файла func.h

#ifndef FUNC\_H

#define FUNC\_H

//Task 1

unsigned int strlen(const char \*str);

//Task 2

char \*find(const char\* str, const char\* substr);

//Task 3

void delete(char \*str, const char \*substr);

//Task 4

void strcpy(char \*dest, int len, const char\* src);

//Task 5

void strcat(char \*dest, int len, const char \*first, const char \*second);

//Task 6

\_Bool insert(const char \*src, const char \*str,int index,char \*dest,int len);

#endif

Листинг 8 – Исходный код файла task7.c

#include <stdio.h>

#include <math.h>

#include <conio.h>

#include <stdlib.h>

#include "func.h"

int main(void)

{

unsigned int len = 0,destlen = 0,index = 0 ;

char \*point;

\_Bool err;

\_Bool b = 1;

while (b==1)

{

printf("Menu:\n");

printf("1. Task 1\n");

printf("2. Task 2\n");

printf("3. Task 3\n");

printf("4. Task 4\n");

printf("5. Task 5\n");

printf("6. Task 6\n");

printf("7. Exit \n");

printf("Select Task number: ");

switch (\_getch())

{

case '1':

{

printf("Task 1:\n");

char str[] = " MAX ";

printf("str = ");

scanf("%s", &str);

len = strlen(str);

printf("strlen = ");

printf("%i", len);

printf("\n");

break;

}

case '2':

{

printf("Task 2:\n");

char str[] = " MAX ";

printf("str = ");

scanf("%s", &str);

printf("\n");

char substr[] = " MAX ";

printf("substr = ");

scanf("%s", &substr);

printf("\n");

point = find(str,substr);

point == 0 ? printf("result = 0"):printf("result = ");printf("%s",substr);

printf("\n");

break;

}

case '3':

{

printf("Task 3:\n");

char str[] = " MAX ";

printf("str = ");

scanf("%s", &str);

printf("\n");

char substr[] = " MAX ";

printf("substr = ");

scanf("%s", &substr);

printf("\n");

delete(str,substr);

printf("strdelete = ");

printf("%s",str);

printf("\n");

break;

}

case '4':

{

printf("Task 4:\n");

int sizeBuffer;

char str[] = " MAX ";

printf("str = ");

scanf("%s", &str);

printf("\n");

printf("sizeBuffer = ");

scanf("%i", &sizeBuffer);

printf("\n");

char buff[sizeBuffer];

strcpy(buff,sizeBuffer,str);

printf("Buffer = ");

printf("%s", buff);

printf("\n");

break;

}

case '5':

{

printf("Task 5:\n");

int sizeBuffer;

printf("sizeBuffer = ");

scanf("%i", &sizeBuffer);

printf("\n");

char buff[sizeBuffer];

char str1[] = " MAX ";

char str2[] = " MAX ";

printf("str1 = ");

scanf("%s", &str1);

printf("\n");

printf("str2 = ");

scanf("%s", &str2);

printf("\n");

strcat(buff,sizeBuffer,str1,str2);

printf("buff = ");

printf("%s", buff);

printf("\n");

break;

}

case '6':

{

printf("Task 6:\n");

char dest[100];

destlen = 100;

char str1[] = " MAX ";

char str2[] = " MAX ";

printf("str1 = ");

scanf("%s", &str1);

printf("\n");

printf("str2 = ");

scanf("%s", &str2);

printf("\n");

printf("index = ");

scanf("%i", &index);

err = insert(str1,str2,index,dest,destlen);

printf("result = ");

err ? printf("%s\n",dest):printf("Erorr\n" );

break;

}

case '7':

{

b = 0;

break;

}

default:

{

printf("\nInvalid value entered. Try again.\n");;

break;

}

}

}

system("pause");

return 0;

}

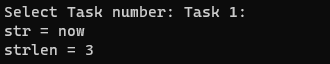


Рисунок 1 - Пример работы функции strlen в файле task1.c

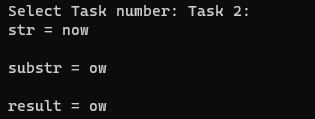


Рисунок 2 - Пример работы функции find в файле task2.c

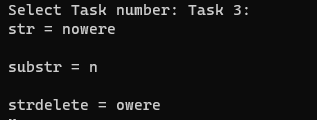


Рисунок 3 - Пример работы функции delete в файле task3.c

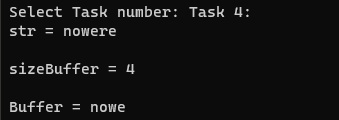


Рисунок 4 - Пример работы функции strcpy в файле task4.c

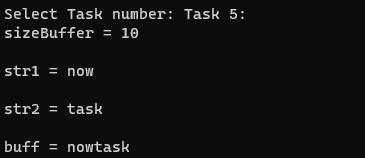


Рисунок 5 - Пример работы функции strcat в файле task5.c

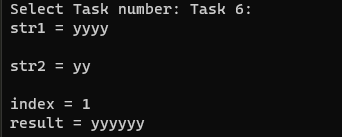


Рисунок 6 - Пример работы функции insert в файле task6.c

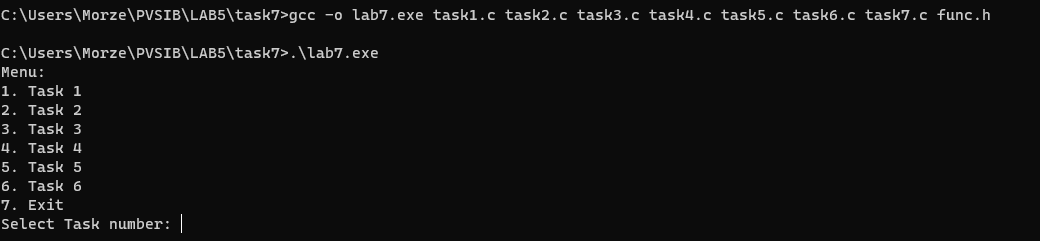


Рисунок 6 - Пример работы функции main в файле task7.c

<https://github.com/Pavel-Plotnikov-BSU1801/Laboratornai-5-Plotnikov-P.S.-BSU1801->

**Вывод**

В данной работе были разобраны работы со строками. Так же были изучены стандартные функции и написаны их аналоги.