ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ  
Ордена Трудового Красного Знамени федеральное государственное   
бюджетное учреждение высшего образования

Московский технический университет связи и информатики

МТУСИ

Кафедра информационной безопасности

Лабораторная работа

Вариант 2

Выполнил студент группы БИН1703 ОТФ-2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бен Режеб Софиэн

Старший преподаватель кафедры ИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Барков

**Практикум №5**  
**Строки**

**Цель практикума:** Овладеть навыками работы со строками в языке программирвоания 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.

***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.

***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.

***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.

***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.

***Задание 6***

Написать функцию вставки подстроки в строку. Функция принимает исходнную строку, строку для вставки, индекс, определяющий место вставки, буфер для размещения новой строки и размер буфера. Если сумма длин исходной строки и строки для встивки 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.

***Задание 7***

Написать функцию main, демонстрирующую работу ранее разработанных функций

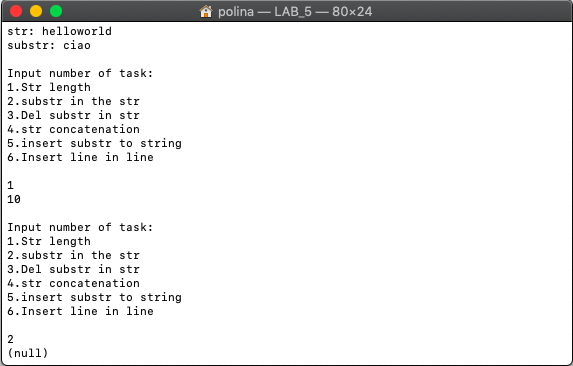


Рисунок 1

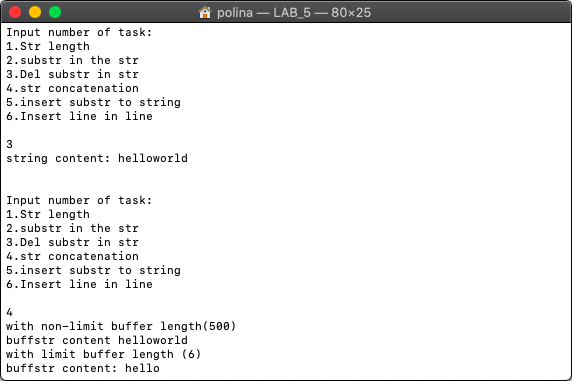


Рисунок 2

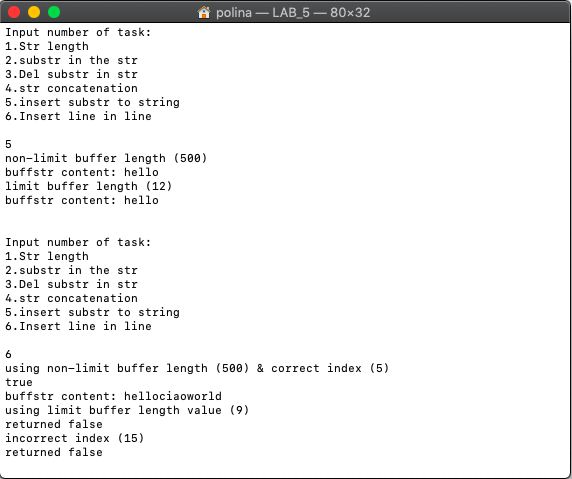


Рисунок 3

Листинг 1 — Исходный файл LAB\_5.c

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

int my\_strlen(const char\* c\_str)

{

int count=0;

while(\*c\_str++) ++count;

return count;

}

char \*find( char \*c\_str, char \*substr) {

int lenс\_Str = my\_strlen(c\_str);

int lenSub = my\_strlen(substr);

for (int i = 0; i <= lenс\_Str - lenSub; ++i) {

int j;

for (j = 0; c\_str[i + j] == substr[j]; ++j);

if (j - lenSub == 1 && i == lenс\_Str - lenSub) return &c\_str[i] ;

}

return 0;

}

void delete( char \*c\_str, char \*substr)

{

int i = 0;

int j;

int substr\_length = my\_strlen(substr);

char \*index = find(c\_str, substr);

if (index == NULL)

return;

int f = 1;

for (int i = 0; i < f; ++i)

{

++f;

index[i] = index[substr\_length + i];

if (index[substr\_length + i] == 0)

break;

}

for (j = 1; j < substr\_length; j++)

{

index[i + j] = 0;

}

}

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

{

int i = 0;

for (i = 0; i < len - 1; i++)

{

dest[i] = src[i];

if (src[i] == 0)

return;

}

dest[i] = 0;

}

void my\_strcat(char\* dest, int len, const char\* first, const char\* second) {

int i;

dest[len - 1] = '\0';

for (i = 0; dest[i] != '\0'; ++i) {

if (first[i] != '\0') {

dest[i] = first[i];

}

else

{

dest[i] = second[i];

}

}

}

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

{

int i = 0;

int j = 0;

if ((my\_strlen(src) + my\_strlen(str) + 1 > len) || (index < 0 || index > my\_strlen(src) - 1))

return 0;

while (i < index)

{

dest[i] = src[i];

i++;

}

while (str[j] != 0)

{

dest[i] = str[j];

i++;

j++;

}

j = index;

while (src[j] != 0)

{

dest[i] = src[j];

i++;

j++;

}

dest[i] = 0;

return 1;

}

int main()

{

char c\_str[100];

char substr[100];

char buffstr[500];

printf("str: ");

gets(c\_str);

printf("substr: ");

gets(substr);

int f = 1;

for (int i = 0; i < f; ++i)

{

f=6;

printf("\nInput number of task:\n1.Str length\n2.substr in the str\n3.Del substr in str\n4.str concatenation\n5.insert substr to string\n6.Insert line in line\n\n");

switch (getchar())

{

case '1':

printf("%d\n", my\_strlen(c\_str));

getchar();

continue;

case '2':

getchar();

continue;

case '3':

delete(c\_str, substr);

printf("string content: %s\n\n", c\_str);

getchar();

continue;

case '4':

my\_strcpy(buffstr, 200, c\_str);

printf("with non-limit buffer length(500)");

printf("\nbuffstr content %s", buffstr);

my\_strcpy(buffstr, my\_strlen(c\_str) / 2 + 1, c\_str);

printf("\nwith limit buffer length (%d)", my\_strlen(c\_str) / 2 + 1);

printf("\nbuffstr content: %s\n\n", buffstr);

getchar();

continue;

case '5':

my\_strcat(buffstr, 500, c\_str, substr);

printf("non-limit buffer length (500)");

printf("\nbuffstr content: %s", buffstr);

my\_strcat(buffstr, my\_strlen(c\_str) + my\_strlen(substr) / 2, c\_str, substr);

printf("\nlimit buffer length (%d)", my\_strlen(c\_str) + my\_strlen(substr) / 2);

printf("\nbuffstr content: %s\n\n", buffstr);

getchar();

continue;

case '6':

printf("using non-limit buffer length (500) & correct index (%d)", my\_strlen(c\_str) / 2);

printf("\n%s", insert(c\_str, substr, my\_strlen(c\_str) / 2, buffstr, 500) ? "true" : "false");

printf("\nbuffstr content: %s", buffstr);

printf("\nusing limit buffer length value (%d)", (my\_strlen(c\_str) + my\_strlen(substr)) / 4 \* 3);

printf("\nreturned %s", insert(c\_str, substr, my\_strlen(c\_str) / 2, buffstr, (my\_strlen(c\_str) + my\_strlen(substr)) / 4 \* 3) ? "true" : "false");

printf("\nincorrect index (%d)", my\_strlen(c\_str) + 5);

printf("\nreturned %s\n\n", insert(c\_str, substr, my\_strlen(c\_str) + 5, buffstr, 500) ? "true" : "false");

continue;

}

}

}