|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |
| Институт искусственного интеллекта |
| Кафедра Технологий Искусственного Интеллекта |

Практическая работа № 12

по дисциплине

«Процедурное программирование»

Обучающийся: Погосян С. А.

Группа: КВБО-07-23

Руководитель *Яковлев Д. А*

Москва 2023

**Тема: «Строки и символьные массивы в языке Си»**

**Цель лабораторной работы:**

Целью данной лабораторной работы освоить на практике работу со строками и символьными массивами.

**Описание работы:**

**Научиться работать со строками и проработать щнания в области применения динамической типизации.**

**Задание:**

1. Функция выравнивает строку по правому краю до заданной длины.

**Код программы:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#define STRING "pervanne69"**

**int to\_space(int n, char \*str, char \*str2) {**

**int index = 0;**

**if (n <= strlen(STRING)) {**

**for (int i = strlen(STRING) - n; i < strlen(STRING); ++i) {**

**str2[index] = str[i];**

**index += 1;**

**}**

**} else {**

**for (int i = 0; i < n - strlen(STRING); ++i) {**

**str2[i] = NULL;**

**}**

**for (int i = n - strlen(STRING); i < n; ++i) {**

**str2[i] = str[index];**

**index += 1;**

**}**

**}**

**return 0;**

**}**

**int main() {**

**char \*str = malloc(sizeof(char) \* (strlen(STRING) + 1));**

**char \*str2;**

**int n;**

**int size\_res = 0;**

**strcpy(str, STRING);**

**printf("%s\n", str);**

**printf("Set the string result length:\n");**

**scanf("%d", &n);**

**if (n <= strlen(STRING)) {**

**size\_res = strlen(STRING) - (strlen(STRING) - n);;**

**} else {**

**size\_res = strlen(STRING) + (n - strlen(STRING));**

**}**

**str2 = malloc(sizeof(char) \* (size\_res + 1));**

**printf("\nResult:\n");**

**to\_space(n, str, str2);**

**for (int i = 0; i < size\_res + 1; ++i) {**

**printf("%c", str2[i]);**

**}**

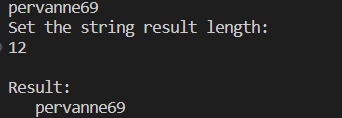
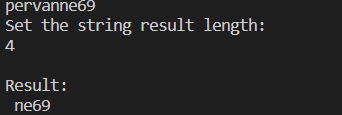
**free(str);**

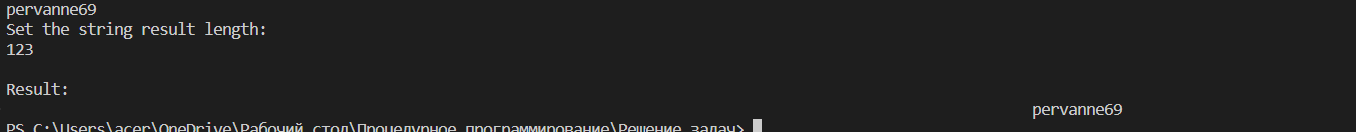
**free(str2);**

**return 0;**

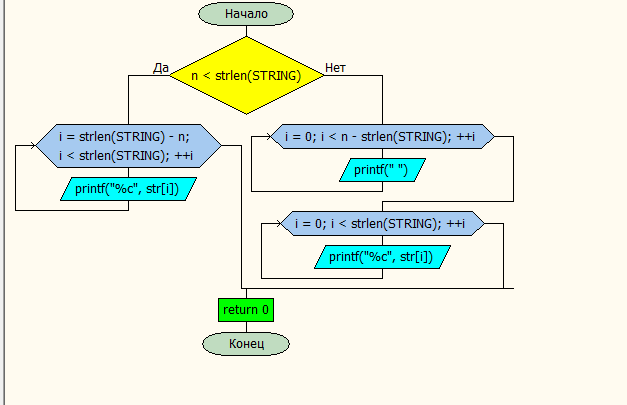
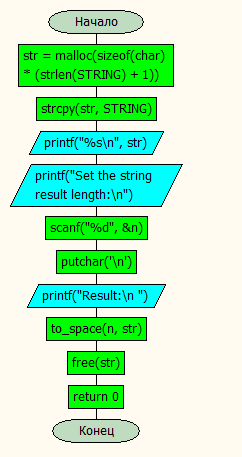
**}**

**Результат работа программы:**

** **

****

**Алгоритм в виде блок-схемы:**

****

**Задание:**

1. Функция дописывает одну строку у начало другой

**Код программы:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define STRING1 "pervanne69"

#define STRING2 "cool"

int fasten\_strings(char \*str1, char \*str2, char \*str3) {

    for (int i = 0; i < strlen(STRING1); ++i) {

        str3[i] = str1[i];

    }

    for (int i = 0; i < strlen(STRING2); ++i) {

        str3[strlen(STRING1) + i] = str2[i];

    }

    return 0;

}

int main() {

    char \*str1 = malloc(sizeof(char) \* (strlen(STRING1) + 1));

    char \*str2 = malloc(sizeof(char) \* (strlen(STRING2) + 1));

    char \*str3 = malloc(sizeof(char) \* (strlen(STRING1) + strlen(STRING2) + 1));

    strcpy(str1, STRING1);

    strcpy(str2, STRING2);

    printf("%s\n", str1);

    printf("%s\n", str2);

    fasten\_strings(str1, str2, str3);

    for (int i = 0; i < strlen(STRING1) + strlen(STRING2); ++i) {

        printf("%c", str3[i]);

    }

    putchar('\n');

    free(str1);

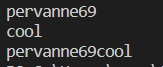
    free(str2);

    free(str3);

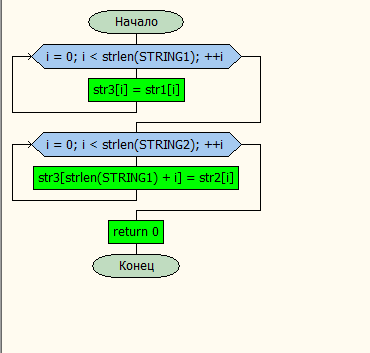
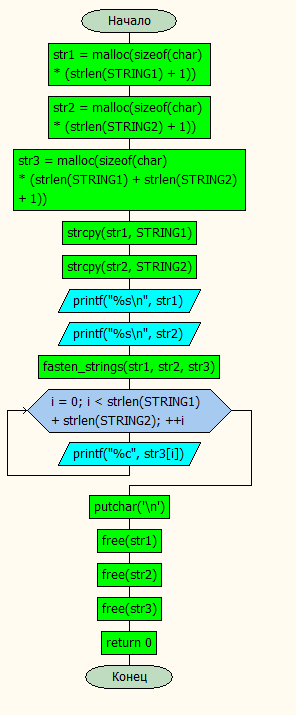
    return 0;

}

**Результат работы программы:**

****

**Алгоритм в виде блок-схемы:**

**** ****

**Контрольные вопросы:**

1. **Как организовать динамическое выделение памяти?**

Память для массива целых чисел в нашей работе не выделяется на этапе компиляции, так что нам достаточно объявить в программе только переменную - указатель на начало массива:

\*\*ar

-указатель на начало массива строк \* mas

Размерность массива определяется при выполнении программы, так что для ее сохранения нужна только одна переменная:

n

Элементы массива будут храниться в динамической памяти (куча), поэтому по окончании работы память занятую указателями необходимо освободить.

1. **Почему необходимо избегать утечки памяти?**

Утечка является проблемой, так как она блокирует ресурсы памяти, что со временем приводит к ухудшению производительности системы.

1. **В чем разница между функциями malloc() и calloc()?**

Функция malloc() выделяет область памяти заданного размера и возвращает указатель на эту область. Функция calloc() выделяет область памяти заданного размера и заполняет ее нулями.

1. **Смысл использования free()?**

Освобождение памяти из кучи

1. **Смысл работы функции realloc()?**

Функция realloc выполняет перераспределение блоков памяти.  
Размер блока памяти, на который ссылается параметр ptrmem изменяется на size байтов. Блок памяти может уменьшаться или увеличиваться в размере.

1. **Основные преимущества динамического выделения памяти?**

Контроль памяти**,** c помощью динамического выделения памяти под указатели строк можно размещать свободные массивы. Свободным называется двухмерный массив (матрица), размер строк которого может быть различным.