Лабораторная работа №4

Введение в функции.

Базовая работа со строками (однобайтовыми).

Комплект 1: Введение в функции

Задание 1.1

Задача:

1.1: Создайте две функции, которые вычисляют факториал числа:

- функцию, которая вычисляет факториал, используя цикл;
- функцию, которая вычисляет факториал, используя рекурсивный вызов самой себя.

Продемонстрируйте работу обеих функций.

Список идентификаторов:

Имя	Смысл	Тип
x	Вводимое число	int
res	Факториал вводимого числа	int

Код программы:

```c
#include <stdio.h>

int fact1(int num)
{
    if (num <= 1) return 1;
    else return num*fact1(num - 1);
}

int fact2(int num)
{
    double result = 1;
    for (int i = 1; i <= num; i++)
    {
        result = result*i;
    }

    return(result);
}

int main()
{
    int x, res;
    printf("x= ");
    scanf("%d", &x);
    res = fact1(x);
    printf("fun1: %d! = %d\n", x, res);

    res = fact2(x);
```

```
    printf("fun2: %d! = %d", x, res);

    return 0;
}
```

Результат выполнения программы:

```
x=10
 fun1: 10! = 3628800
fun2: 10! = 3628800
Process finished with exit code 0
```

Задание 1.2

Задача:

1.2: Объявите указатель на массив типа **int** и динамически выделите память для 12-ти элементов. Напишите функцию, которая поменяет значения чётных и нечётных ячеек массива.

Список идентификаторов:

Имя	Смысл	Тип
mass	Вводимый с клавиатуры массив	double
num	Количество элементов массива	int
i	Параметр цикла	int

Код программы:

```
#include <stdio.h>
#include <stdlib.h>
void swap(double *mass)
{
    double buffer;
    for (int i = 0; i < 11; i++)
    {
        if (i%2==0)
        {
            buffer = mass[i];
            mass[i] = mass[i+1];
            mass[i+1] = buffer;
        }
    }
}
int main()
{
    double *mass;
    int num=12;
    mass = (double *) malloc(num * sizeof(double));
    for (int i = 0; i < num; i++) {
```

```
        printf("Enter element of massive: \n");
        scanf("%lf", &mass[i]);
    }
    printf("Mass: ");
    swap(mass);
    for(int i = 0; i < num; i++)
    {
        printf("%.2lf ", mass[i]);
    }
    free(mass);
    return 0;
}
```

Результат выполнения программы:

```
Enter element of massive:
1
Enter element of massive:
2
Enter element of massive:
3
Enter element of massive:
4
Enter element of massive:
5
Enter element of massive:
6
Enter element of massive:
7
Enter element of massive:
8
Enter element of massive:
9
Enter element of massive:
10
Enter element of massive:
11
Enter element of massive:
12
Mass: 2.00 1.00 4.00 3.00 6.00 5.00 8.00 7.00 10.00 9.00 12.00 11.00
Process finished with exit code 0
```

Задание 1.3

Задача:

1.3: Создать две основные функции:

- функцию для динамического выделения памяти под двумерный динамический массив типа **double** — матрицу;
- функцию для динамического освобождения памяти под двумерный динамический массив типа **double** — матрицу.

Создать две вспомогательные функции:

- функцию для заполнения матрицы типа **double**;
- функцию для распечатки этой матрицы на экране.

Продемонстрировать работу всех этих функций в своей программе.

Список идентификаторов:

Имя	Смысл	Тип
rows	Количество строк матрицы	int
columns	Количество столбцов матрицы	int
mass	Двумерный массив (матрица)	double

Код программы:

```c
#include <stdio.h>
#include <stdlib.h>


double ** matrix( int columns,int rows) {
    double **mass;
    mass = (double **) malloc(rows * sizeof(double *));
    for (int i = 0; i < rows; i++) {
        mass[i] = (double *) malloc(columns * sizeof(double *));
    }
    return(mass);
}

void matrixDeploy(double **mass, int rows, int columns)
{
    for (int i = 0; i < rows; i++)
    {
        double a;
        for (int j = 0; j < columns; j++)
        {
            printf("Enter element mass[%d][%d]=\n", i+1, j+1);
            scanf("%lf",&a);
            mass[i][j]=a;
        }
    }
}

void matrixApp(double **mass, int rows, int columns)
{
    printf("\n");
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < columns; j++)
        {
            printf("%.1lf ", mass[i][j]);
        }
        printf("\n");
    }
}

void matrixFr(double **mass, int rows, int columns)
{
    for (int i = 0; i < rows; i++)
    {
        free(mass[i]);
    }
    free(mass);
    printf("Success");
}
```

```c
int main()
{
    int rows, columns;
    printf("Enter number of rows \n");
    scanf("%d", &rows);
    printf("Enter number of columns \n");
    scanf("%d", &columns);
    double **mass;
    mass = matrix(columns,rows);
    matrixDeploy(mass,rows,columns);
    matrixApp(mass,rows,columns);
    matrixFr(mass,rows,columns);

    return 0;
}
```

Результат выполнения программы:

```
Enter number of rows
3
Enter number of columns
3
Enter element mass[1][1]=
1
Enter element mass[1][2]=
5
Enter element mass[1][3]=
2
Enter element mass[2][1]=
64
Enter element mass[2][2]=
7.2
Enter element mass[2][3]=
14.3
Enter element mass[3][1]=
14.87
Enter element mass[3][2]=
1
Enter element mass[3][3]=
9

1.0 5.0 2.0
64.0 7.2 14.3
14.9 1.0 9.0
Success
Process finished with exit code 0
```

Задание 1.4

Задача:

4.4: Создать функцию, которая вычисляет векторное произведение двух векторов в декартовых координатах, используя указатели на соответствующие массивы.

Список идентификаторов:

Имя	Смысл	Тип
size	Размер массивов	int
vector1	Массив для вектора 1	double
vector2	Массив для вектора 2	double
result	Векторное произведение	double
i	Параметр цикла	int

Код программы:

```
#include <stdio.h>
#include <stdlib.h>
double * product(int size,double *vector1,double *vector2)
{
    double *res;
    res = (double *) malloc(size * sizeof(double));
    res[0] = vector1[1]*vector2[2]-vector1[2]*vector2[1];
    res[1] = vector1[0]*vector2[2]-vector1[2]*vector2[0];
    res[2] = vector1[0]*vector2[1]-vector1[1]*vector2[0];
    return (res);
}
int main()
{
    int size = 3;
    double *vector1;
    vector1 = (double *) malloc(size * sizeof(double));
    for (int i = 0; i < size; i++) {
        printf("\nEnter vector1: ");
        scanf("%lf", &vector1[i]);
    }
    double *vector2;
    vector2 = (double *) malloc(size * sizeof(double));
    for (int i = 0; i < size; i++) {
        printf("\nEnter vector2: ");
        scanf("%lf", &vector2[i]);
    }
    double *result;
    result = (double *) malloc(size * sizeof(double));
    for (int i =0; i<size; i++)
    {
        result[i]=product(size,vector1,vector2)[i];
    }
    printf("\nvectorRes: {%.2lf,%.2lf,%.2lf}",
result[0],result[1],result[2]);
    free(result);
    free(vector1);
    free(vector2);
    return 0;
}
```

Результат выполнения программы:

```
Enter vector1:1

Enter vector1:2

Enter vector1:3

Enter vector2:3

Enter vector2:2

Enter vector2:1

vectorRes: {-4.00,-8.00,-4.00}
Process finished with exit code 0
```

Комплект 2: Базовые операции со строками

Задание 2.1

Задача:

2.1: Создайте новую программу, где с клавиатуры вводится строка некоторой длины порядка 10 *латинских* символов (не используйте кириллицу) в классическую строку языка C, которая имеет вид массива `char my_string[MY_SIZE]`. `MY_SIZE` определите с помощью директивы `#define`. Значение `MY_SIZE` должно превышать длину вводимой строки с некоторым разумным запасом. Другие строки в этой задаче можете создавать либо также как статические массивы, либо как динамические массивы, но не забывайте освобождать от динамически выделенную память с помощью функции
`void free(void* ptr);` .
Выполните следующие действия и распечатайте результаты:

1. Вычислите длину строки `my_string`, используя цикл `for` и тот факт, что в языкеC такие строки имеют в конце специальный нулевой символ конца строки, представленный escape-последовательностью `'\0'` (`'...'` — это тип `char`).
2. Сделайте тоже самое, что в пункте 1, но создайте указатель на начало вашей строки и используйте операцию инкремента ++.
3. Используйте функции
`size_t strlen(const char* str);` или

`size_t strnlen (const char *string, size_t maxlen);` или
`size_t strnlen_s(const char *str, size_t strsz);` для получения размера строки в виде значения `size_t` (псевдоним `unsigned int`, спецификатор форматирования — `"%zu"`). Убедитесь, что ваш компилятор явно работает с опцией `-std=c11` или с опцией для более позднего стандарта языка для поддержки функции `strnlen_s`.
4. Создайте вторую строку (второй массив) и скопируйте в неё строку `my_string`, используя функцию
`char *strcpy(char *dest, const char *src);` или
`char *strncpy (char *dest, const char *src, size_t n);`.
5. Создайте ещё две строки какого-либо размера и задайте их прямо в коде без клавиатуры. Сделайте конкатенацию этих двух строк, используя
`char *strcat(char *dest, const char *src);` или
`char *strncat(char *dest, const char *src, size_t n);`. Первую строку трактуйте как `dest` (destination) и подберите размер этого массива с запасом.
6. Сравните две новые строки, заданные в коде строковыми литералами, используя функцию
`int strcmp(const char *lhs, const char *rhs);` или
`int strncmp (const char *s1, const char *s2, size_t n)`.
7. Задайте прямо в коде строку, в которой есть только латинские символы в верхнем и нижнем регистре. Переведите строку полностью в нижний регистр и отдельно полностью в верхний регистр. Распечатайте каждый результат отдельно. Найдите сигнатуры подходящих функций (`tolower` и `toupper`), изучив *базовые* однобайтовые строковые функции по ссылке `https://en.cppreference.com/w/c/string/byte`.

Список идентификаторов:

Имя	Смысл	Тип
size	Вспомогательная переменная	int
i	Параметр цикла	int
size1	Вспомогательная переменная	unsigned int
my_string	Символьная строка	char
p_string	Указатель на my_string	char
my_string1	Символьная строка	char
my_string2	Символьная строка	char
my_string3	Символьная строка	char
my_string4	Символьная строка	char
my_string5	Символьная строка	char
my_string6	Символьная строка	char

Код программы:

```
#include<stdio.h>
#include <string.h>
#include <ctype.h>
#define MY_SIZE 15

int main()
{
    char my_string[MY_SIZE];
    printf("Enter 10 symbols: ");
    gets(my_string);
    int size = 0;
    for(int i = 0; i<MY_SIZE; i++)
    {
        if (my_string[i] == '\0')
        {
            break;
        }
        else
        {
            size++;
        }
    }
    printf("1)Size = %d\n", size);
    printf("\n");
    char *p_string = NULL;
    p_string = &my_string[0];
    size = 0;
    for(int i = 0; i<MY_SIZE; i++)
    {
        if (*p_string == '\0')
        {
            break;
        }
        else
        {
            size++;
            p_string++;
        }
    }
```

```
    printf("2)Size = %d\n", size);
    printf("\n");
    unsigned int size1;
    size1 = strlen(my_string);
    printf("3)Size = %ld\n", size1);
    printf("\n");
    char my_string1[MY_SIZE];
    strcpy(my_string1, my_string);
    printf("string2: %s\n",my_string1);
    printf("\n");
    char my_string2[15] = {"qwerty"};
    printf("string part 1: %s\n", my_string2);
    char my_string3[6] = {"uiop"};
    printf("string part 2: %s\n", my_string3);
    printf("Destination = %s\n",strcat(my_string2, my_string3));
    printf("\n");
    char my_string4[15] = {"Hello world"};
    printf("string 1: %s\n", my_string4);
    char my_string5[15] = {"Let's begin"};
    printf("string 2: %s\n", my_string5);
    if (strcmp(my_string4,my_string5)==0)
    {
        printf("They are similar\n");
    }
    else
    {
        printf("They are different\n");
    }
    printf("\n");
    char my_string6[] = {"AbCdEfGhLmnbUibn"};
    printf("string: %s\n", my_string6);
    for(int i = 0; i < strlen(my_string6); i++)
    {
        my_string6[i]=tolower(my_string6[i]);
    }
    printf("string: %s\n", my_string6);
    for(int i = 0; i < strlen(my_string6); i++)
    {
        my_string6[i]=toupper(my_string6[i]);
    }
    printf("string: %s\n", my_string6);
}
```

Результат выполнения программы:

```
Enter 10 symbols:12rt15fcy9
 1)Size = 10

2)Size = 10

3)Size = 10

string2: 12rt15fcy9

string part 1: qwerty
string part 2: uiop
Destination = qwertyuiop

string 1: Hello world
string 2: Let's begin
They are different

string: AbCdEfGhLmnbUibn
string: abcdefghlmnbuibn
string: ABCDEFGHLMNBUIBN

Process finished with exit code 0
```

Задание 2.2

Задача:

2.2: Конвертируйте введённые заданные как строки: число с плавающей точкой (`double`) и целое число (`int`) в значения типа `double` и `int`, используя функциями `atof` и `atoi`. См. документацию по ссылке `https://en.cppreference.com/w/c/string/byte`.

Список идентификаторов:

Имя	Смысл	Тип
my_string1	Символьная строка	char
my_string2	Символьная строка	char

Код программы:

```c
#include<stdio.h>
#include <stdlib.h>


int main()
{
    char my_string1[25];
    printf("1)Enter string of 25 or less symbols: \n");
    gets(my_string1);
```

```
    printf("Int part: %d \n", atoi(my_string1));
    char my_string2[25];
    printf("2)Enter string of 25 or less symbols: \n");
    gets(my_string1);
    printf("Double part: %lf \n", atof(my_string1));
}
```

Результат выполнения программы:

```
1)Enter string of 25 or less symbols:
12dr1531r
Int part: 12
2)Enter string of 25 or less symbols:
1151.11613.geglkv
Double part: 1151.116130

Process finished with exit code 0
```

Задание 2.3

Задача:

2.3: Создайте строку от 10 до 20 символов, используя только цифры, латинский буквы в разных регистрах пробельные символы и символы пунктуации. Организуйте цикл, где каждый символ подробно тестируется функциями типа `int is*(/*... */)` (например — `isdigit`, `ispunct`). См. документацию по ссылке `https://en.cppreference.com/w/c/string/byte`. Оформите распечатку информации по каждому символу в виде списка на экране, чтобы можно было прочесть информацию о том что представляет из себя каждый символ (своими словами, в свободной форме). Постарайтесь использовать только латиницу.

Список идентификаторов:

Имя	Смысл	Тип
my_string	Символьная строка	char
i	Параметр цикла	int

Код программы:

```
#include<stdio.h>
#include <string.h>
#include <ctype.h>
#define MY_SIZE 20

int main()
{
```

```
    char my_string[MY_SIZE];
    printf("Enter 20 or less symbols: ");
    gets(my_string);
    for(int i = 0; i<MY_SIZE; i++)
    {
        if (my_string[i] == '\0')
        {
            break;
        }
        printf("%d) ",(i+1));
        if (isalnum(my_string[i]))
        {
            printf("letter or digit; ");
        }

        if (isalpha(my_string[i]))
        {
            printf("letter, ");
        }

        if (islower(my_string[i]))
        {
            printf("lowercase, ");
        }

        if (isupper(my_string[i]))
        {
            printf("uppercase, ");
        }

        if (isdigit(my_string[i]))
        {
            printf("digit, ");
        }

        if (isxdigit(my_string[i]))
        {
            printf("xdigit, ");
        }

        if (iscntrl(my_string[i]))
        {
            printf("ASCII, ");
        }

        if (isgraph(my_string[i]))
        {
            printf("graphical, ");
        }

        if (isspace(my_string[i]))
        {
            printf("space, ");
        }

        if (isblank(my string[i]))
        {
            printf("blank, ");
        }

        if (isprint(my_string[i]))
        {
            printf("printing, ");
        }
```

```
        if (ispunct(my_string[i]))
        {
            printf("punctuation, ");
        }
        printf("IT IS ALL\n");
    }
}
```

Результат выполнения программы:

```
Enter 20 or less symbols:12tYb0km12.d':)
 1) letter or digit; digit, xdigit, graphical, printing, IT IS ALL
2) letter or digit; digit, xdigit, graphical, printing, IT IS ALL
3) letter or digit; letter, lowercase, graphical, printing, IT IS ALL
4) letter or digit; letter, uppercase, graphical, printing, IT IS ALL
5) letter or digit; letter, lowercase, xdigit, graphical, printing, IT IS ALL
6) letter or digit; letter, uppercase, graphical, printing, IT IS ALL
7) letter or digit; letter, lowercase, graphical, printing, IT IS ALL
8) letter or digit; letter, lowercase, graphical, printing, IT IS ALL
9) letter or digit; digit, xdigit, graphical, printing, IT IS ALL
10) letter or digit; digit, xdigit, graphical, printing, IT IS ALL
11) graphical, printing, punctuation, IT IS ALL
12) letter or digit; letter, lowercase, xdigit, graphical, printing, IT IS ALL
13) graphical, printing, punctuation, IT IS ALL
14) graphical, printing, punctuation, IT IS ALL
15) graphical, printing, punctuation, IT IS ALL

Process finished with exit code 0
```