

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»
(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова)

Факультет Среднего профессионального образования
Дисциплина Основы алгоритмизации и программирования
наименование дисциплины

ЛАБОРАТОРНАЯ РАБОТА №4

номер (при наличии)

Массивы

при наличии указать тему лабораторной работы и (или) номер варианта

ОБУЧАЮЩИЙСЯ

группы 09С51
подпись _____ фамилия и инициалы Куманов Д.В.

дата сдачи

ПРОВЕРИЛ
Шарипова Э.Р.

подпись _____ фамилия и инициалы

Оценка / балльная оценка _____

дата проверки

г. Санкт-Петербург
2025 г.

Требования к выполнению.

В отчете необходимо:

1. Для каждого задания указать условие его выполнения.
2. К каждому заданию приложить:
 - 2.1 Текст программы.
3. Результаты тестирования необходимо демонстрировать через скриншоты.
4. Все переменные в программах должны быть названы понятно и читаемо для преподавателя. Например, переменные можно называть как: var1, var2, max, min, res, count, и т.д.
5. Выполнять задания необходимо строго по своим вариантам.

Вариант 1

Статический массив:

Задание 1.

Напишите программу, которая удаляет все отрицательные элементы массива, сжимая его. Остальные элементы должны быть перемещены в начало массива, а в конце должны оставаться только нули. Выведите массив до и после удаления отрицательных элементов.

Код программы:

```
#include <stdio.h>

void removeNegatives(int array[], int length) {
    int writeIndex = 0;
    for (int readIndex = 0; readIndex < length; readIndex++) {
        if (array[readIndex] >= 0) {
            array[writeIndex++] = array[readIndex];
        }
    }
    while (writeIndex < length) {
        array[writeIndex++] = 0;
    }
}

int main() {
    int numbers[] = {3, -1, 4, -5, 6, 0, -2};
    int size = sizeof(numbers) / sizeof(numbers[0]);

    printf("Массив до удаления отрицательных элементов:\n");
    for (int i = 0; i < size; i++) {
        printf("%d ", numbers[i]);
    }
    printf("\n");

    removeNegatives(numbers, size);

    printf("Массив после удаления отрицательных элементов:\n");
    for (int i = 0; i < size; i++) {
        printf("%d ", numbers[i]);
    }
    return 0;
}
```

На рисунке 1 представлен результат программы.

```
Массив до удаления отрицательных элементов:
3 -1 4 -5 6 0 -2
Массив после удаления отрицательных элементов:
3 4 6 0 0 0 0
```

Рисунок 1 – Результат программы

Задание 2.

Напишите программу, которая создает одномерный массив случайных чисел.

Реализуйте функцию для нахождения минимального и максимального значения в массиве.

Динамический массив (количество строк и столбцов вводятся с клавиатуры):

Код программы:

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

// Функция для нахождения минимального и максимального значения в массиве
void findMinMax(int array[], int length, int *minValue, int *maxValue) {
    *minValue = array[0];
    *maxValue = array[0];
    for (int i = 1; i < length; i++) {
        if (array[i] < *minValue) *minValue = array[i];
        if (array[i] > *maxValue) *maxValue = array[i];
    }
}

int main() {
    int length;
    printf("Введите размер одномерного массива: ");
    scanf("%d", &length);

    int *array = malloc(length * sizeof(int));
    if (array == NULL) {
        printf("ошибка выделения памяти.\n");
        return 1;
    }

    srand(time(NULL));
    for (int i = 0; i < length; i++) {
        array[i] = rand() % 100; // заполнение случайными числами от 0 до 99
    }

    printf("Сгенерированный массив:\n");
    for (int i = 0; i < length; i++) {
        printf("%d ", array[i]);
    }
    printf("\n");

    int minValue, maxValue;
    findMinMax(array, length, &minValue, &maxValue);
    printf("Минимальное значение: %d\n", minValue);
    printf("Максимальное значение: %d\n", maxValue);

    // Работа с динамическим двумерным массивом
    int rows, cols;
    printf("Введите количество строк динамического массива: ");
    scanf("%d", &rows);
```

```

printf("Введите количество столбцов динамического массива: ");
scanf("%d", &cols);

// Выделение памяти под двумерный динамический массив
int **matrix = malloc(rows * sizeof(int *));
if (matrix == NULL) {
    printf("Ошибка выделения памяти.\n");
    free(array);
    return 1;
}
for (int i = 0; i < rows; i++) {
    matrix[i] = malloc(cols * sizeof(int));
    if (matrix[i] == NULL) {
        printf("Ошибка выделения памяти.\n");
        for (int k = 0; k < i; k++) {
            free(matrix[k]);
        }
        free(matrix);
        free(array);
        return 1;
    }
}

printf("Введите элементы динамического массива:\n");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        scanf("%d", &matrix[i][j]);
    }
}

printf("Введенный динамический массив:\n");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        printf("%d ", matrix[i][j]);
    }
    printf("\n");
}

// Очистка памяти
for (int i = 0; i < rows; i++) {
    free(matrix[i]);
}
free(matrix);
free(array);

return 0;
}

```

На рисунке 2 представлен результат программы.

```

Введите размер одномерного массива: 4
Сгенерированный массив:
96 9 14 57
Минимальное значение: 9
Максимальное значение: 96
Введите количество строк динамического массива: 4
Введите количество столбцов динамического массива: 4
Введите элементы динамического массива:
1 2 3 4 5

сч
Введенный динамический массив:
1 2 3 4
5 13828288 0 552530685
13843928 13828288 51512066 34603536
13843928 13828288 50397953 50397953

```

Рисунок 2 – Результат программы

Задание 3.

Напишите программу, которая принимает квадратную матрицу и возвращает сумму элементов по главной диагонали.

Код программы:

```
#include <stdio.h>

int main() {
    int size;
    printf("Введите размер квадратной матрицы: ");
    scanf("%d", &size);

    int matrix[size][size];

    printf("Введите элементы матрицы:\n");
    for (int row = 0; row < size; row++) {
        for (int col = 0; col < size; col++) {
            scanf("%d", &matrix[row][col]);
        }
    }

    int diagonalSum = 0;
    for (int i = 0; i < size; i++) {
        diagonalSum += matrix[i][i];
    }

    printf("Сумма элементов главной диагонали: %d\n", diagonalSum);
    return 0;
}
```

На рисунке 3 представлен результат программы.

```
Введите размер квадратной матрицы: 2
Введите элементы матрицы:
1 2 3 4
Сумма элементов главной диагонали: 5
```

Рисунок 3 – Результат программы

Задание 4.

Напишите программу, которая принимает квадратную матрицу и возвращает её транспонированную версию (поменяйте строки и столбцы местами).

Код программы:

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

int main() {
    int size;
    printf("Введите размер квадратной матрицы: ");
    scanf("%d", &size);
```

```

if (size > 1000) { // огроменная слишком,
    printf("Слишком большой размер.\n");
    return 1;
}

// Динам массивы и выдел памяти
int **matrix = malloc(size * sizeof(int *));
int **transpose = malloc(size * sizeof(int *));

for (int i = 0; i < size; i++) {
    matrix[i] = malloc(size * sizeof(int));
    transpose[i] = malloc(size * sizeof(int));
}

printf("Введите элементы матрицы:\n");
for (int row = 0; row < size; row++) {
    for (int col = 0; col < size; col++) {
        scanf("%d", &matrix[row][col]);
    }
}

// Транспонир
for (int row = 0; row < size; row++) {
    for (int col = 0; col < size; col++) {
        transpose[col][row] = matrix[row][col];
    }
}

printf("Транспонированная матрица:\n");
for (int row = 0; row < size; row++) {
    for (int col = 0; col < size; col++) {
        printf("%d ", transpose[row][col]);
    }
    printf("\n");
}

// Освобождение памяти
for (int i = 0; i < size; i++) {
    free(matrix[i]);
    free(transpose[i]);
}
free(matrix);
free(transpose);

return 0;
}

```

На рисунке 4 представлен результат программы.

```

Ведите размер квадратной матрицы: 3
Ведите элементы матрицы:
1 2 3 4 5 6
Ъъ
Транспонированная матрица:
1 4 0
2 5 48
3 6 0

```

Рисунок 4 – Результат программы