

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОССУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
**БЕЛГОРОДСКИЙ ГОССУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. Шухова**
(БГТУ им. В. Г. Шухова)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И
УПРАВЛЯЮЩИХ СИСТЕМ

Кафедра программного обеспечения вычислительной техники и
автоматизированных систем

Отчет

По учебно-ознакомительной практике

Выполнил: студент группы КБ-232

Башков Михаил Антонович

(подпись студента)

Проверил: ассистент Новожен Н.В.

(подпись руководителя практики)

Оценка: _____

Белгород 2024 г.

Оглавление

Компьютерная практика

1. Тема 1. Линейные алгоритмы
2. Тема 2. Разветвляющиеся алгоритмы
3. Тема 3. Циклические и итерационные алгоритмы
4. Тема 4. Простейшие операции над массивами
5. Тема 5. Векторы и матрицы
6. Тема 6. Линейный поиск
7. Тема 7. Арифметика
8. Тема 8. Геометрия и теория множеств
9. Тема 9. Линейная алгебра и сжатие информации
10. Тема 10. Алгоритмы обработки символьной информации
11. Тема 11. Аналитическая геометрия
12. Тема 12. Кривые второго порядка на плоскости
13. Тема 13. Графическое решение систем уравнений
14. Тема 14. Плоскость в трехмерном пространстве
15. Тема 15. Поверхность второго порядка в трехмерном пространстве

Задания к работе

Тема 1.

Угол α задан в градусах, минутах и секундах. Найти его величину в радианах (с максимально возможной точностью).

Тема 2.

Заданы три числа: a , b , c . Определить, могут ли они быть сторонами треугольника, и если да, то определить его тип: равносторонний, равнобедренный, разносторонний.

Тема 3.

Численно убедиться, является ли заданная функция $y = f(x)$ чётной или нечётной на заданном отрезке $-a \leq x \leq a$. Учесть погрешность вычислений и возможные точки разрыва функции.

Тема 4.

В массиве $C(n)$ подсчитать количество отрицательных и сумму положительных элементов.

Тема 5.

Строки матрицы $A(m, n)$ заполнены не полностью: в массиве $L(m)$ указано количество элементов в каждой строке. Переслать элементы матрицы построчно в начало одномерного массива $T(m \cdot n)$ и подсчитать их количество.

Тема 6.

Седловой точкой в матрице называется элемент, являющийся одновременно наибольшим в столбце и наименьшим в строке. Седловых точек может быть несколько. В матрице $A(m, n)$ найти все седловые точки либо установить, что таких точек нет.

Тема 7.

Натуральное число в p -ичной системе счисления задано своими цифрами, хранящимися в массиве $K(n)$. Проверить корректность такого представления и перевести число в q -ичную систему (возможно, число слишком велико, чтобы получить его внутреннее представление; кроме того, $p \leq 10$, $q \leq 10$).

Тема 8.

Заяц, хаотично прыгая, оставил след в виде замкнутой самопересекающейся ломаной, охватывающей территорию его владения (отрезки ломаной заданы длиной прыжка и его направлением по азимуту). Найти площадь минимального по площади выпуклого многоугольника, описанного вокруг этой территории.

Тема 9.

Выполнить операцию транспонирования прямоугольной матрицы $A(m, n)$, $m \neq n$, не выделяя дополнительного массива для хранения результата. Матрицу представить в виде одномерного массива.

Тема 10.

Текст записан одной длинной строкой. Признаком красной строки служит символ $\$$. Переформатировать текст в 60-символьные строки, формируя абзацы.

Тема 11.

Построить прямую параллельную оси абсцисс (Ox) и пересекающую ось ординат (Oy) в точке $A(0, 2)$ в диапазоне $x \in [-3; 3]$ с шагом $\Delta = 0.5$.

Тема 12.

Построить верхнюю часть параболы $y^2 = x$ при $0 \leq x \leq 4$ с шагом $\Delta = 0.25$.

Тема 13.

$$\begin{cases} y = \frac{2}{x} + 2 \\ z = x^2 + 1 \end{cases}$$

в диапазоне $0.2 \leq x \leq 3$, с шагом $\Delta 0.1$.

Тема 14.

Построить плоскость, параллельную плоскости Oxy и пересекающую ось Oz в точке $M(0, 0, 2)$, при $0 \leq x \leq 6$ с шагом $\Delta = 0.5$ и $0 \leq y \leq 6$ с шагом $\Delta = 1$.

Тема 15.

Построить верхнюю часть эллипсоида, заданного уравнением $\frac{x^2}{9} + \frac{y^2}{4} + z^2 = 1$, лежащую в диапазоне $-3 \leq x \leq 3$, $-2 \leq y \leq 2$ с шагом $\Delta = 0.5$ для обеих переменных.

Основная часть (выполнение заданий)

Задание 1

Тема 1. Линейные алгоритмы

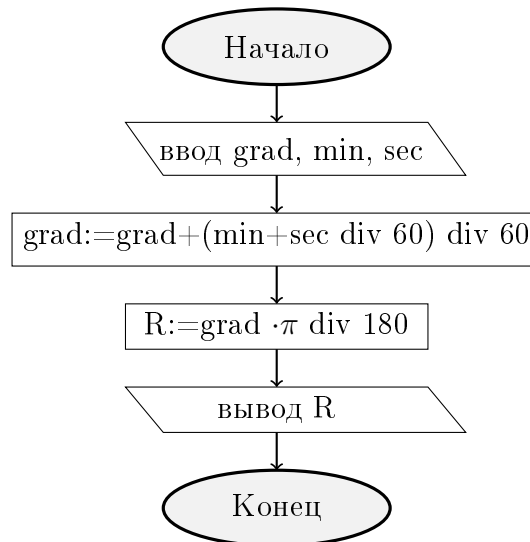
Угол α задан в градусах, минутах и секундах. Найти его величину в радианах (с максимально возможной точностью)

Математическое обоснование и словесное описание

Чтобы перевести угол, заданный в градусах, минутах и секундах в радианы, сначала необходимо градусы, минуты и секунды перевести в градусы.

$degrees^{\circ} = degrees^{\circ} + \frac{minutes}{60} + \frac{second}{3600}$ А потом переводим градусы в радианы руководствуясь данной формулой $R = \frac{degrees^{\circ} \cdot \pi}{180}$

Блок-схема



Код программы

```
double zadanie1(double grad, double min, double sec){  
    return (grad + (min + sec / 60) / 60) * 3.14 / 180;  
}
```

Тестовые данные

Ввод	Вывод
3, 58, 12	0.069254
30, 30, 30	0.532201

Задание 2

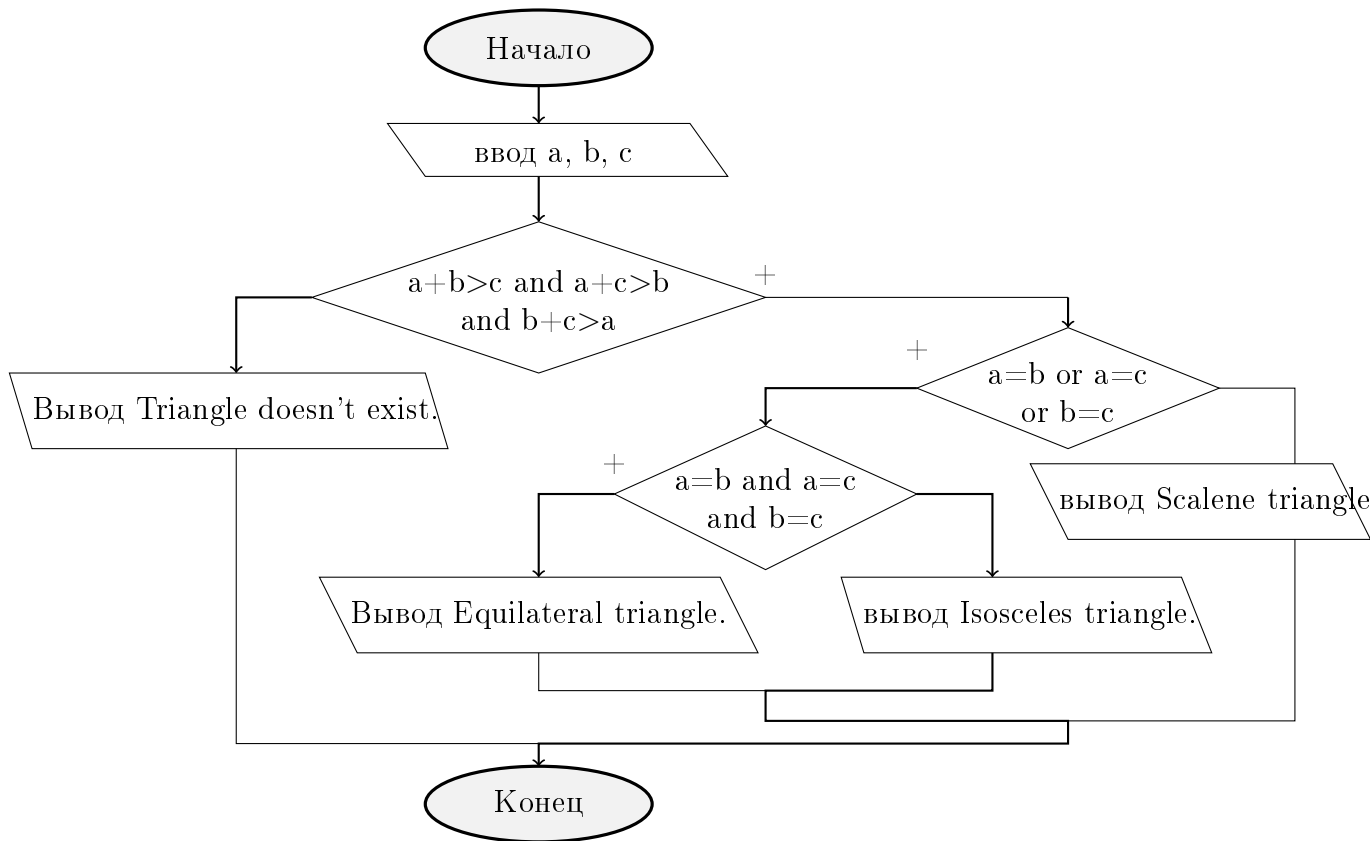
Тема 2. Разветвляющиеся алгоритмы

Заданы три числа: a , b , c . Определить, могут ли они быть сторонами треугольника, и если да, то определить его тип: равносторонний, равнобедренный, разносторонний.

Математическое обоснование и словесное описание

Если стороны треугольника соответствуют условиям $a + b > c$ и $a + c > b$ и $b + c > a$, то он существует. Если $a = b$ или $b = c$ или $a = c$, то он равнобедренный. Если $a = b$ и $b = c$ и $a = c$ то он равносторонний. Если треугольник существует и не выполняется ни одно из условий, то он разносторонний.

Блок-схема



Код программы

```
void zadanie2(int a, int b, int c) {  
    if (a + b > c && a + c > b && b + c > a)  
        if (a == b || a == c || b == c)  
            if (a == b && a == c && b == c)  
                printf("Equilateral triangle.\n");  
            else  
                printf( "Isosceles triangle.\n");  
        else  
            printf( "Scalene triangle.\n");  
    else  
        printf( "Triangle doesn't exist.\n");  
}
```

Тестовые данные

Ввод	Вывод
30,30,30	Equilateral triangle.
30,30,20	Isosceles triangle.

Задание 3

Тема 3. Циклические и итерационные алгоритмы

Численно убедиться, является ли заданная функция $y = f(x)$ чётной или нечётной на заданном отрезке $-a \leq x \leq a$. Учесть погрешность вычислений и возможные точки разрыва функции.

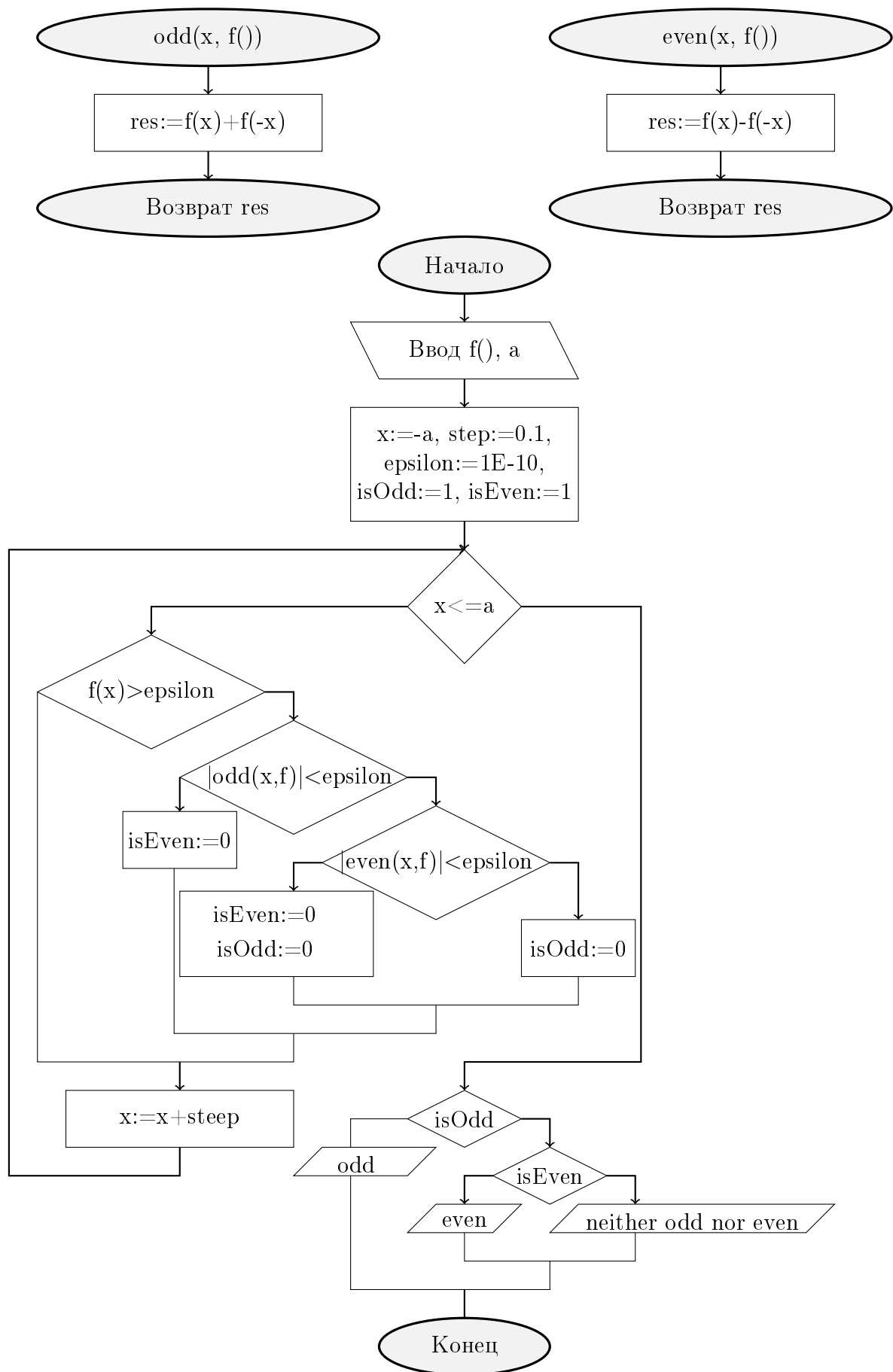
Математическое обоснование и словесное описание

Нечётными и чётными называются функции, обладающие симметрией относительно изменения знака аргумента. Если функция нечетная, то выполняется равенство:

$f(x) + f(-x) = 0$. Если функция четная, то выполняется равенство: $f(x) - f(-x) = 0$.

Пройдем от $-a$ до a с шагом $\Delta = 0.1$ и определим четность или нечетность функции.

Блок-схема



Код программы

```
/* проверка на нечетность. Возвращает ~0 если нечетная */
double odd(double x, double (*f)(double)){
    return f(x) + f(-x);
}

/* проверка на четность. Возвращает ~0 если четная */
double even(double x, double (*f)(double)){
    return f(x) - f(-x);
}

double fabs(double a){
    if(a < 0)
        return -a;
    return a;
}

/* общая проверка, вывод результата */
void zadanie3(double(*f)(double), double a){
    double x = -a, step = 0.1, epsilon = 1E-10;
    int isOdd = 1, isEven = 1;

    while(x <= a){
        if(f(x) > epsilon){
            if(fabs(odd(x,f)) < epsilon)
                isEven *= 0;
            else if(fabs(even(x,f)) < epsilon)
                isOdd *= 0;
            else{
                isEven *= 0;
                isOdd *= 0;
            }
        }
        x += step;
    }
}
```

```
if(isOdd)
    printf("odd");
else if (isEven)
    printf("even");
else
    printf("neither odd nor even");
}
```

Тестовые данные

Ввод	Вывод
sin, 50	odd
cos, 5	even

Задание 4

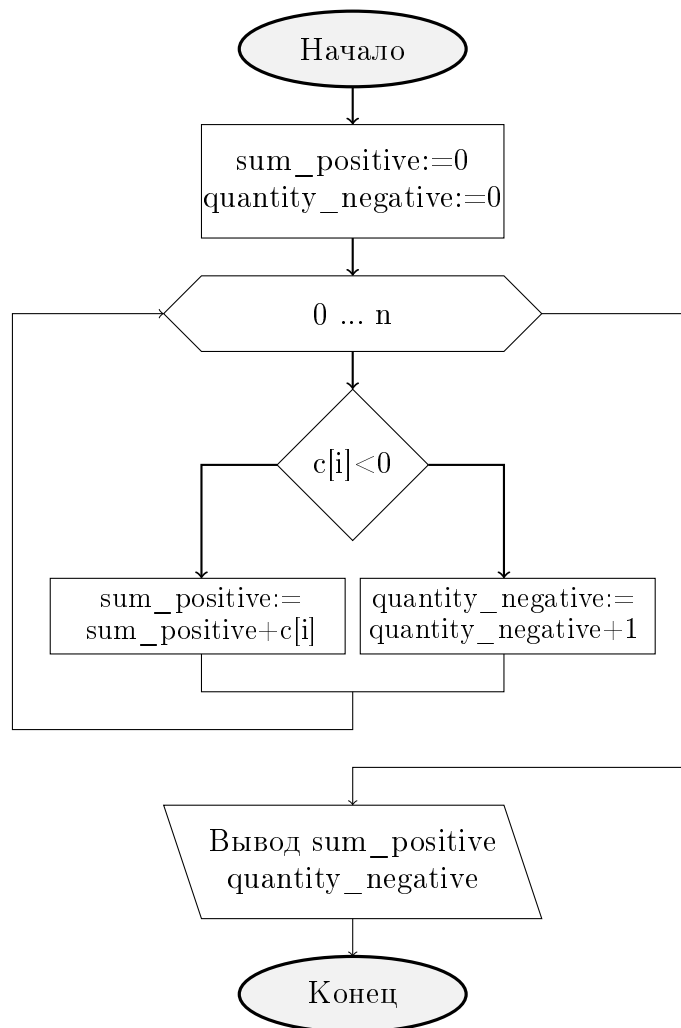
Тема 4. Простейшие операции над массивами

В массиве $C(n)$ подсчитать количество отрицательных и сумму положительных элементов.

Математическое обоснование и словесное описание

Пройдем последовательно по массиву $C(n)$ и если элемент ≥ 0 то будем прибавлять к счетчику $sum_positive$, если нет, то прибавим 1 к $quantity_negative$.

Блок-схема



Код программы

```
void zadanie4(int *c, int n) {  
    int sum_positive = 0;  
    int quantity_negative = 0;  
    for(int i = 0; i < n; i++) {  
        if(c[i] < 0)  
            quantity_negative++;  
        else  
            sum_positive += c[i];  
    }  
  
    printf("\n%d - sum positive elements\n", sum_positive);  
    printf("%d - quantity negative elements\n", quantity_negative);  
}
```

Тестовые данные

Ввод	Вывод
{1,2,3,4,4,0,-1,-2,13}, 9	27 - sum positive 2 - quantity negative
{-1,-2,-3,-4,-5,-6,7,8,9}, 9	24 - sum positive 6 - quantity negative

Задание 5

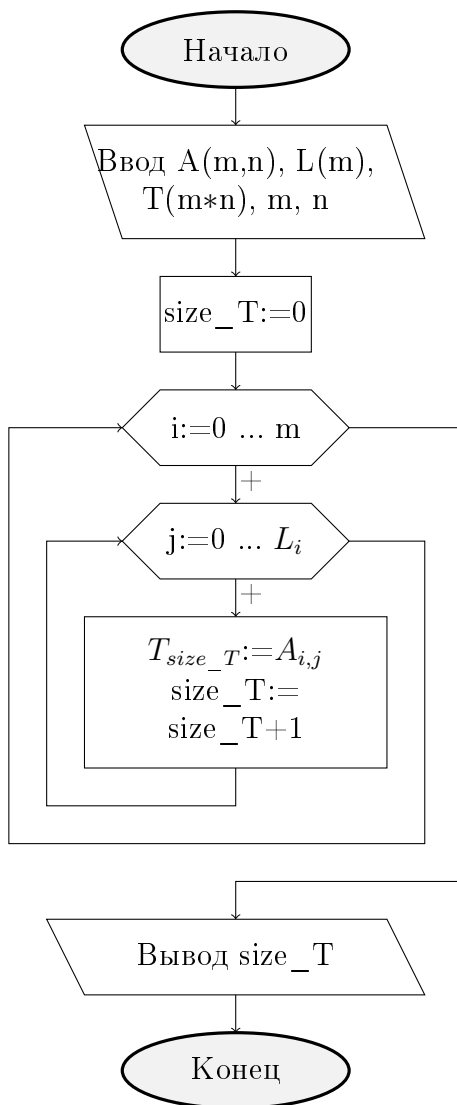
Тема 5. Векторы и матрицы

Строки матрицы $A(m, n)$ заполнены не полностью: в массиве $L(m)$ указано количество элементов в каждой строке. Переслать элементы матрицы построчно в начало одномерного массива $T(m \cdot n)$ и подсчитать их количество.

Математическое обоснование и словесное описание

Пройдем по столбцам матрицы $A(m, n)$ и по строкам, в соответствии с числом, хранящимся в массиве $L(m)$. Будем добавлять в массив $T(m \cdot n)$ элементы, попутно считая их в счетчике $size_T$.

Блок-схема



Код программы

```
int zadanie5(int **A, int *L, int *T, int m, int n) {  
    int size_T = 0;  
    for(int i = 0; i < m; i++) {  
        for(int j = 0; j < L[i]; j++) {  
            T[size_T++] = A[i][j];  
        }  
    }  
  
    return size_T;  
}
```

Тестовые данные

Ввод	Вывод
A={{1,2,3},{4,5,6},{7,8,9}} L={1,1,2}, T={}, 3, 3	T={1,4,7,8}, 4
A={{1,1,1},{2,2,2},{3,3,3}} L={2,1,0}, T={}, 3, 3	T={1,1,2}, 3

Задание 6

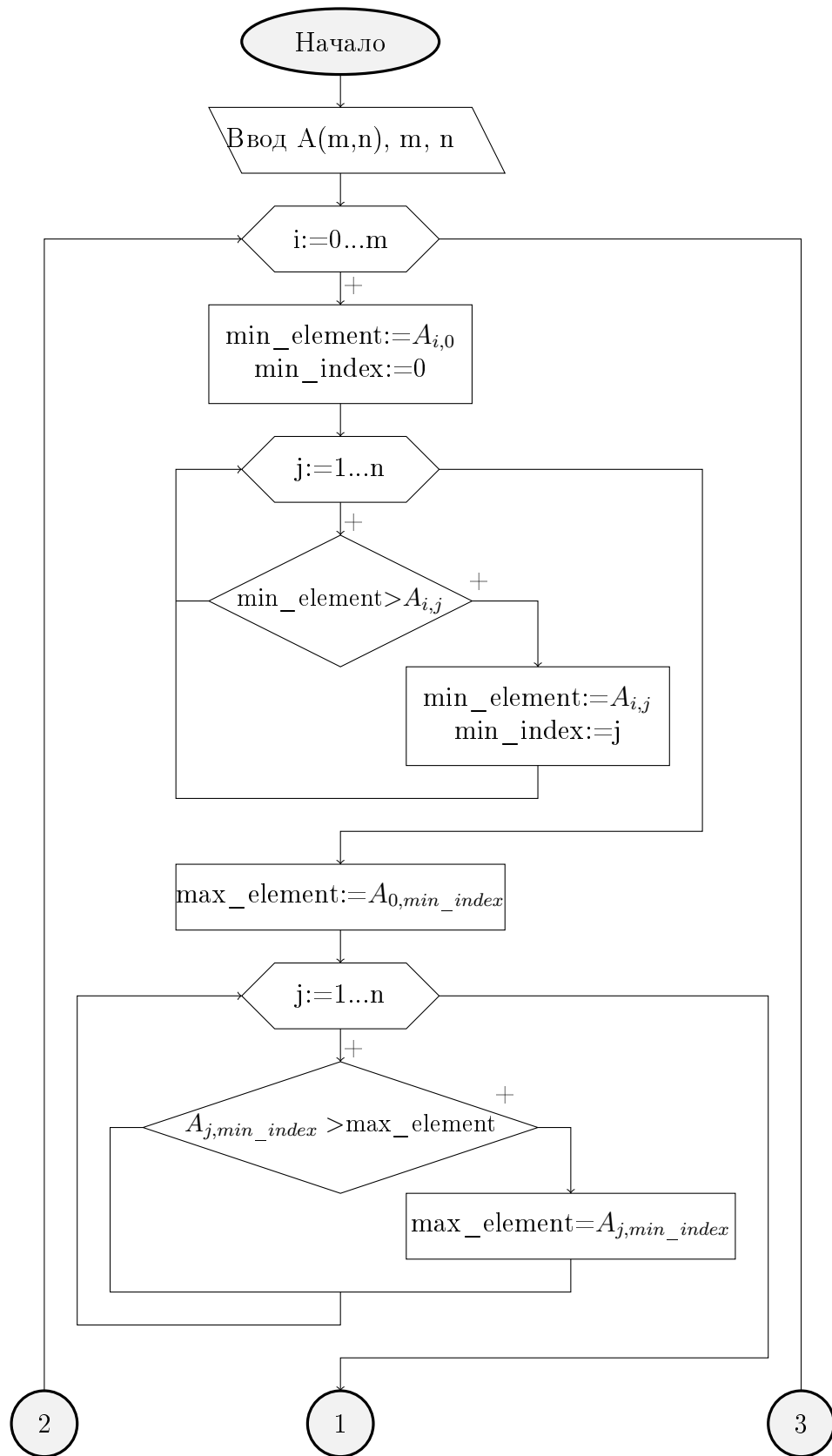
Тема 6. Линейный поиск

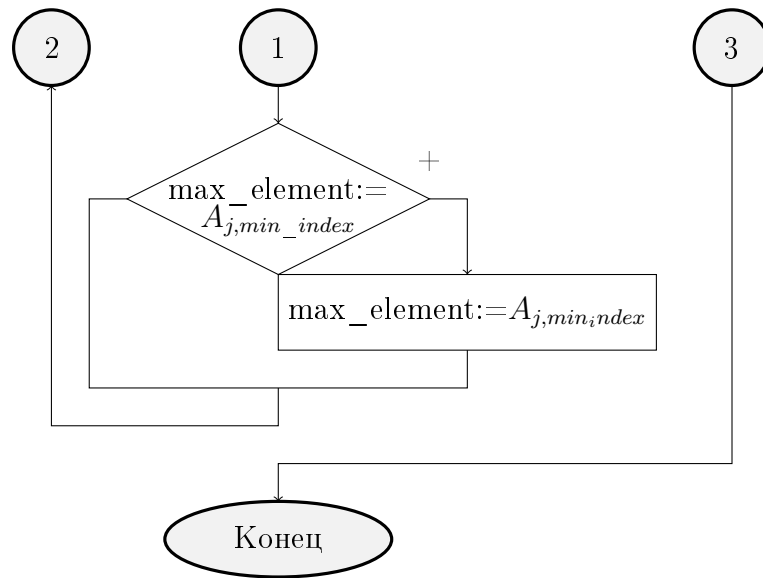
Седловой точкой в матрице называется элемент, являющийся одновременно наибольшим в столбце и наименьшим в строке. Седловых точек может быть несколько. В матрице $A(m, n)$ найти все седловые точки либо установить, что таких точек нет.

Математическое обоснование и словесное описание

Пройдем по столбцам матрицы $A(m, n)$. Пройдем по строкам, найдем минимальный элемент и его индекс. Пройдем по столбцу с минимальным индексом и найдем максимальный элемент. Если максимальный элемент будет равен минимальному, то выводим элемент и координаты, в которых он находится.

Блок-схема





Код программы

```
void zadanie6(int **A, int m, int n) {
    for(int i = 0; i < m; i++) {
        int min_element = A[i][0];
        int min_index = 0;

        for(int j = 1; j < n; j++)
            if(min_element > A[i][j])
                min_element = A[i][j], min_index = j;

        int max_element = A[0][min_index];
        for(int j = 1; j < m; j++)
            if(A[j][min_index] > max_element)
                max_element = A[j][min_index];

        if(max_element == min_element)
            printf("\n(%d) (%d, %d) is saddle point\n", min_element, i, min_index);
    }
}
```

Тестовые данные

Ввод	Вывод
A={{1,2,3},{4,5,6},{7,8,9},{10,11,12}}, 4, 3	(10) (3, 0) is saddle point
A={{1,2,3},{4,5,6},{7,8,9},{1,1,1}}, 4, 3	(7) (2, 0) is saddle point

Задание 7

Тема 7. Арифметика

Натуральное число в p -ичной системе счисления задано своими цифрами, хранящимися в массиве $K(n)$. Проверить корректность такого представления и перевести число в q -ичную систему (возможно, число слишком велико, чтобы получить его внутреннее представление; кроме того, $p \leq 10, q \leq 10$).

Математическое обоснование и словесное описание

Для начала пройдем по массиву $K(n)$ и проверим, $K_i < p$. Если условие не соблюдается, то число некорректно в данной p -ичной системе. Если число корректно, то сначала переведем из p -ичной системы в 10-чную. После переведем это 10-чное число в q -ичную систему. (Для перевода числа из p -ичной системы в 10-чную, воспользуемся формулой $abc_x = (a \cdot x^2 + b \cdot x^1 + c \cdot x^0)_{10}$. Для перевода из 10-чной системы в q -ичную достаточно число в 10-чной системе делить на q и записывать остаток от деления в обратном порядке в число в q -ичной системе.)

Код программы

```
void zadanie7(int *K, int n, int p, int q) {
    int is_correctly = 1;
    for(int i = 0; i < n; i++)
        if(K[i] < 0 && K[i] >= p)
            is_correctly = 0;

    if(is_correctly) {
        int digit = 0;
        for(int i = 0, j = n - 1; i < n; i++, j--) {
            int factor = 1;
            for(int k = 0; k < j; k++)
                factor *= p;
            digit += K[i] * factor;
        }

        int res[n + n];
        int size_res = 0;
        while(digit != 0) {
            res[size_res++] = digit % q;
            digit /= q;
        }
        for(int i = size_res-1; i >= 0; i--)
            printf("%d", res[i]);
        printf("\n");
    }
}
```

Тестовые данные

Ввод	Вывод
K[3]={5,6,7}, 3, 8, 2	101110111
Q[2]={7,7}, 2, 8, 10	63

Задание 8

Тема 8. Геометрия и теория множеств

Заяц, хаотично прыгая, оставил след в виде замкнутой самопересекающейся ломаной, охватывающей территорию его владения (отрезки ломаной заданы длиной прыжка и его направлением по азимуту). Найти площадь минимального по площади выпуклого многоугольника, описанного вокруг этой территории.

Математическое обоснование и словесное описание

Используем алгоритм Грэхема для построения выпуклой оболочки:

1. Найти точку с минимальной y -координатой. Если таких точек несколько, выбрать точку с минимальной x -координатой. Эта точка будет первой вершиной выпуклой оболочки.
2. Отсортировать все остальные точки по углу, образуемому вектором между первой точкой и каждой из остальных точек, и осью x .
3. Добавлять точки в выпуклую оболочку одну за другой в порядке возрастания угла. Если добавление точки приводит к повороту выпуклой оболочки направо, то точка добавляется. В противном случае, точка отбрасывается.

После найдем площадь фигуры с помощью формулы $(x_i \cdot y_j - x_j \cdot y_i)/2$

Код программы

```
#define MAX_POINTS 1000

typedef struct Point {
    double x;
    double y;
} Point;

int compare(const void *a, const void *b) {
    Point *p1 = (Point *)a;
    Point *p2 = (Point *)b;
    return (p1->y - p2->y);
}

int orientation(Point p, Point q, Point r) {
    double val = (q.y - p.y) * (r.x - q.x) - (q.x - p.x) * (r.y - q.y);
    if (val == 0) return 0; // коллинеарный
    return (val > 0) ? 1 : 2; // по часовой стрелке или против часовой стрелки
}

void convexHull(Point points[], int n, Point hull[]) {
    // Найдем самую нижнюю точку
    int ymin = points[0].y, min = 0;
    for (int i = 1; i < n; i++) {
        int y = points[i].y;
        if ((y < ymin) || (ymin == y && points[i].x < points[min].x))
            ymin = points[i].y, min = i;
    }

    // Поместим самую нижнюю точку в первую позицию
    Point temp = points[0];
    points[0] = points[min];
    points[min] = temp;
}
```

```

// Отсортируем n-1 точек относительно первой точки.
qsort(points + 1, n - 1, sizeof(Point), compare);

/*Если две или более точки составляют одинаковый угол с p0,
удалим все, кроме той, которая находится дальше всего от p0.
*/
int m = 1;
for (int i = 1; i < n; i++) {
    while (i < n - 1 && orientation(points[0], points[i], points[i + 1]) == 0)
        i++;

    points[m] = points[i];
    m++;
}

if (m < 3) {
    for (int i = 0; i < m; i++)
        hull[i] = points[i];
    return;
}

int top = 0;
hull[top++] = points[0];
hull[top++] = points[1];
hull[top++] = points[2];

// Обрабатываем оставшиеся n-3 точки
for (int i = 3; i < m; i++) {
    while (top > 1 && orientation(hull[top - 2], hull[top - 1], points[i]) != 2)
        top--;
    hull[top++] = points[i];
}
}

```

```
double shoelace(Point points[], int n) {
    double area = 0;
    for (int i = 0; i < n; i++) {
        int j = (i + 1) % n;
        area += (points[i].x * points[j].y - points[j].x * points[i].y);
    }
    return abs(area) / 2.0;
}
```

```
double zadanie8(Point *points, int n) {
    Point hull[MAX_POINTS];
    convexHull(points, n, hull);

    return shoelace(hull, n);
}
```

Тестовые данные

Ввод	Вывод
points = {{3,1}, {5,2}, {2,5},{1,1}}, 4	8.500000

Задание 9

Тема 9. Линейная алгебра и сжатие информации

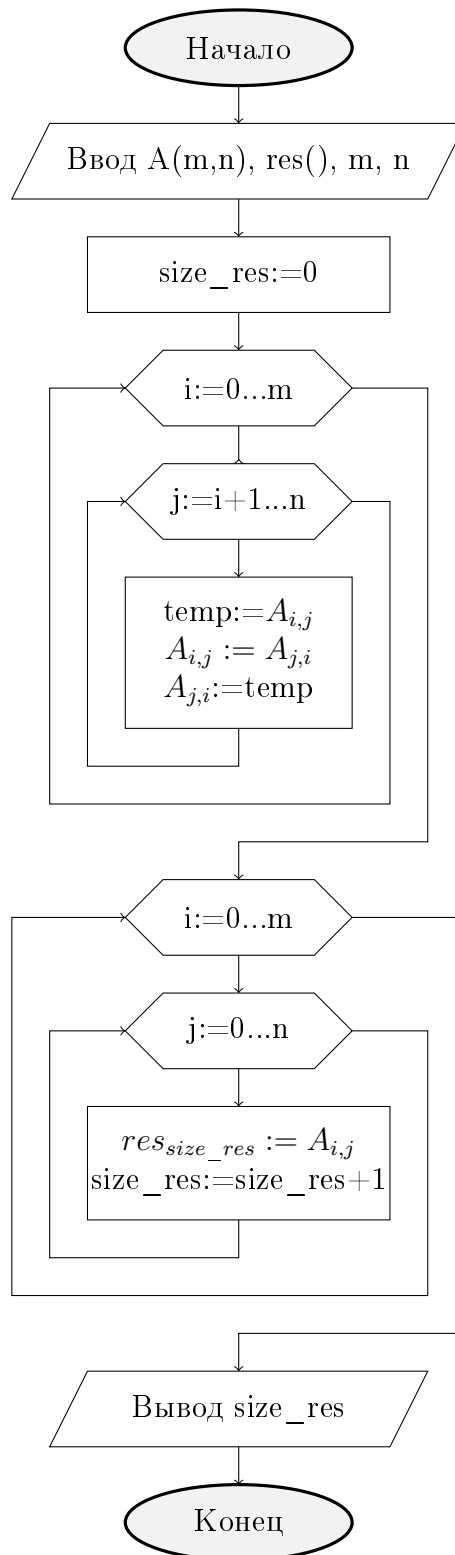
Выполнить операцию транспонирования прямоугольной матрицы $A(m, n)$, $m \neq n$, не выделяя дополнительного массива для хранения результата. Матрицу представить в виде одномерного массива.

Математическое обоснование и словесное описание

Транспонирование матрицы - это операция над матрицей, когда ее строки становятся столбцами с теми же номерами. Если матрица A - это матрица размера $m \times n$, то матрица A^T имеет размер $n \times m$.

Пройдем по столбам матрицы $A(m, n)$ индексом i и по строкам индексом $j = i + 1$ и будем менять $A_{i,j} = A_{j,i}$ и $A_{j,i} = A_{i,j}$. После перенесем матрицу поэлементно в массив *res*.

Блок-схема



Код программы

```
int zadanie9(int **A, int *res, int m, int n) {
    int size_res = 0;
    for(int i = 0; i < m; i++) {
        for(int j = i + 1; j < n; j++) {
            int temp = A[i][j];
            A[i][j] = A[j][i];
            A[j][i] = temp;
        }
    }
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
            res[size_res++] = A[i][j];

    return size_res;
}
```

Тестовые данные

Ввод	Вывод
$A = \{\{0,0\}, \{1,1\}, \{2,2\}\}, T, 3, 2$	$T = \{0,1,0,1,2,2\}$
$A = \{\{0,1\}, \{2,3\}, \{4,5\}\}, T, 3, 2$	$T = \{0,2,1,3,4,5\}$