**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. В.Г.ШУХОВА» (БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной

техники и автоматизированных систем

**Отчет по компьютерной практике**

Вариант 3

Выполнил: ст. группы ПВ-11 Донцов А.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

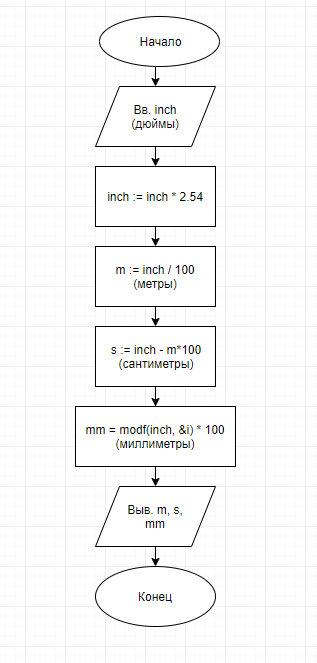
Проверил: Осипов О.В., Бондаренко Т.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись преподавателя)

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Белгород 2018

№1 ЛИНЕЙНЫЕ АЛГОРИТМЫ

Длина отрезка задана в дюймах (1 дюйм = 2,54 см). Перевести

значение длины в метрическую систему, то есть выразить её в метрах,

сантиметрах и миллиметрах. Например, 21 дюйм = 0 м 53 см 3,4 мм

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <math.h>

int main()

{

//переводит дюймы в метры, сантиметры, миллиметры

setlocale(LC\_ALL,"Rus");

int i;

float inch;

printf("Введите размер (в дюймах) = ");

scanf("%f", &inch);

inch = inch \* 2.54;//переводим дюймы в сантиметры

int m,s,mm;

m = inch/100;//метры

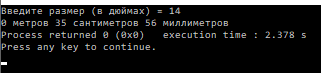
s = inch - m\*100;//сантиметры

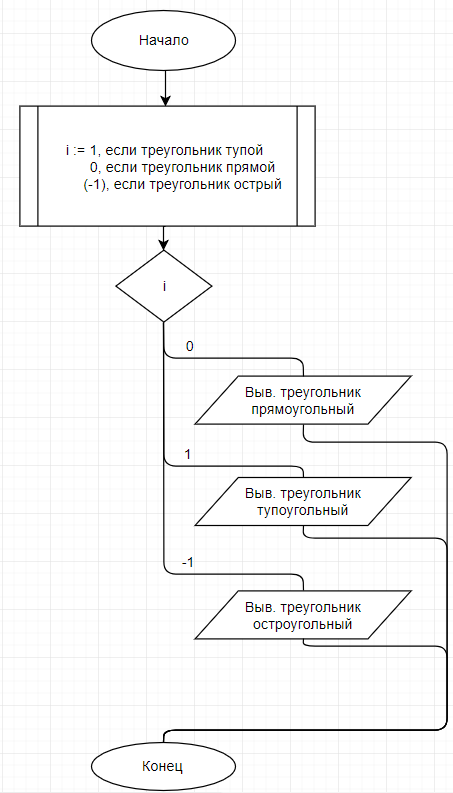
mm = modf(inch, &i) \* 100;//миллиметры

printf("%d метров %d сантиметров %d миллиметров", m, s, mm);

return 0;

}



ТЕМА 2. РАЗВЕТВЛЯЮЩИЕСЯ АЛГОРИТМЫ

Треугольник задан координатами своих вершин на плоскости:

A(x,y), B(x,y), C(x,y). Определить, является он прямо-,

остро- или тупоугольным.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

typedef struct{

float x;

float y;

}point;

void in\_point(point \*);//ввод точек

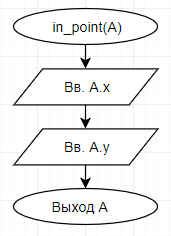
int whi\_trian(point, point, point);//какой треугольник? 1-тупой, 0-прямой, (-1)-острый

float len\_line(point, point);//длинна отрезка

void swap(int \*, int \*);//замена

int main()

{

 setlocale(LC\_ALL, "Rus");//подключаем русский

point A,B,C;//вводим переменные

in\_point(&A);//................

in\_point(&B);//................

in\_point(&C);//................

int i = whi\_trian(A,B,C);

switch (i){

case(0):

printf("Треугольник прямоугольный");

break;

case(1):

printf("Треугольник тупоугольный");

break;

case(-1):

printf("Треугольник остроугольный");

break;

}

return 0;

}

void in\_point(point \*A){//ввод точек

printf("Ввод точки\n");

printf(" Введите координату точки по x = ");

scanf("%f", &A->x);

printf(" Введите координату точки по y = ");

scanf("%f", &A->y);

printf("------------------------------------------\n");

}

int whi\_trian(point A, point B, point C){//какой треугольник? 1-тупой, 0-прямой, (-1)-острый

float hypo, cat1, cat2;

hypo = len\_line(A, B);

cat1 = len\_line(A, C);

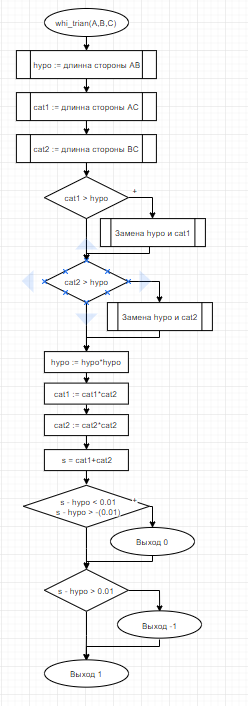
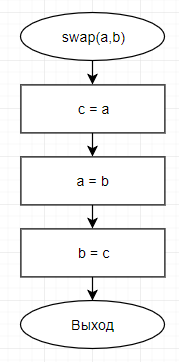
cat2 = len\_line(B, C);

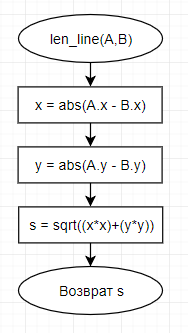
if (cat1 > hypo)

swap(&hypo, &cat1);

if (cat2 > hypo)

swap(&hypo, &cat2);

 hypo \*= hypo;

 cat1 \*= cat1;

cat2 \*= cat2;

float s = cat1+cat2;

if((s - hypo < 0.01)&&(s - hypo > -(0.01)))

return 0;

if(s - hypo > 0.01)

return -1;

else

return 1;

}

float len\_line(point A, point B){//длинна отрезка AB

float s;

float x, y;

x = abs(A.x - B.x);

y = abs(A.y - B.y);

s = sqrt((x\*x)+(y\*y));

return s;

}

void swap(int \*a, int \*b){

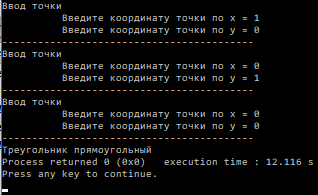
int c;

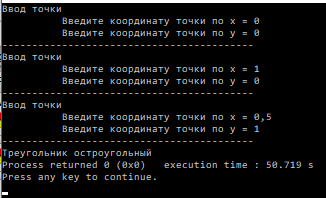
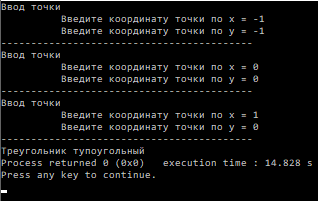
c = \*a;

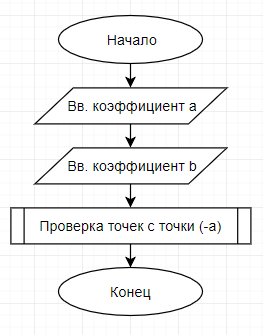
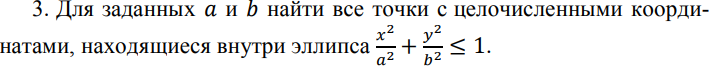
\*a = \*b;

\*b = c;

}





ТЕМА 3. ЦИКЛИЧЕСКИЕ И ИТЕРАЦИОННЫЕ АЛГОРИТМЫ

#include <stdio.h>

#include <stdlib.h>

int a, b;

int main()

{

extern int a, b;

printf("Введите значение a = ");

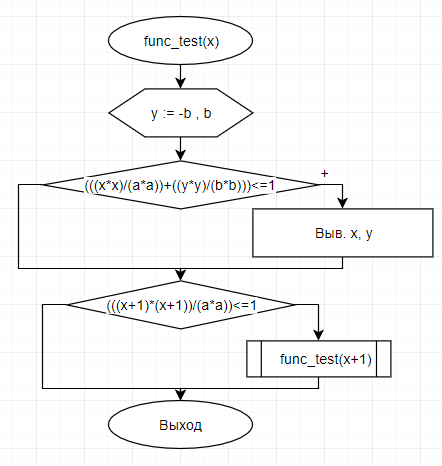
scanf("%d", &a);

printf("Введите значение b = ");

scanf("%d", &b);

func\_test(-a);

return 0;

}

void func\_test(int x){

extern int a,b;

int y;

for(y = -b; y <= b; y++){

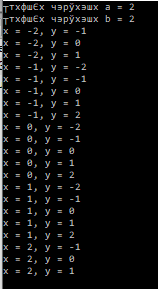
if((((x\*x)/(a\*a))+((y\*y)/(b\*b)))<=1)

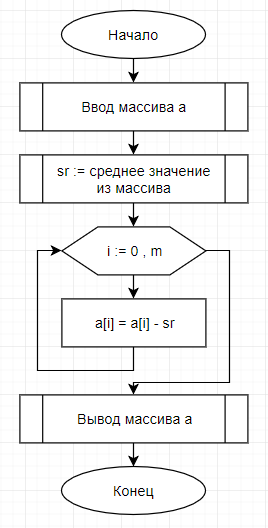
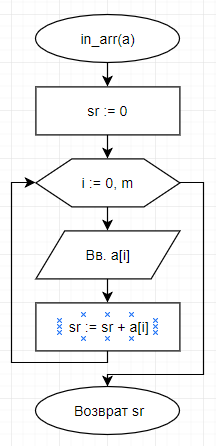
printf("x = %d, y = %d\n", x, y);

}

if((((x+1)\*(x+1))/(a\*a))<=1)

func\_test(x+1);

}

ТЕМА 4. ПРОСТЕЙШИЕ ОПЕРАЦИИ НАД МАССИВАМИ

От каждого из заданных чисел x1, x2, …,xm отнять их среднее арифметическое.

Результаты разместить на месте исходных данных.

#include <stdio.h>

#include <stdlib.h>

#define m 10

//вычитаем среднее значение из всех элементов массива

int in\_arr(int\*);

int out\_arr(int\*);

int main()

{

int a[m], i;

int sr = in\_arr(a) / m;//получение среднего значения

for(i = 0; i < m; i++){

a[i] = a[i] - sr;

}

out\_arr(a);

return 0;

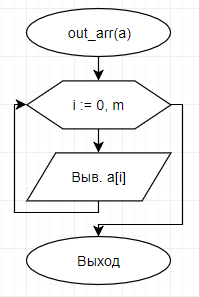
}

int in\_arr(int a[m]){//ввод массива a и возврат суммы чисел sr

int i, sr = 0;

for(i = 0; i < m; i++){

scanf("%d", &a[i]);

 sr = sr + a[i];

}

return sr;

}

int out\_arr(int a[m]){//вывод массива a

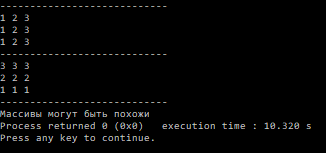
int i;

for(i = 0; i < m; i++)

printf("%d ", a[i]);

}



ТЕМА 5. ВЕКТОРЫ И МАТРИЦЫ

Для двух заданных матриц *А(n,n)* и *В(n,n*) проверить, можно ли получить

вторую из первой применением конечного числа (не более четырёх)

операций транспонирования относительно главной и побочной диагоналей.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#define N 3

void in\_arr(int \*);

int\* sw\_main(int\*);

int\* sw\_sec(int \*);

void swap(int\*, int\*);

void out\_arr(int \*);

int equal\_arr(int \*, int \*);

void copy\_arr(int\*, int\*);

int check\_arr(int \*a, int \*b, int t);

int main()

{

setlocale(LC\_ALL, "Rus");

/\*Ввод матриц a, b\*/

int a[N][N], b[N][N];

printf("----------------------------\n");

in\_arr(a);

in\_arr(b);

check\_arr(a, b, 0);

printf("Массивы не могут быть похожи");

return 0;

}

void in\_arr(int \*a){//Ввод матрицы n\*n

int i;

for(i = 0; i < N\*N; i++)

scanf("%d", &a[i]);

printf("----------------------------\n");

}

int\* sw\_main(int \*a){//работает

int i, j;

for(i = 0; i < N; i++)

for(j = i+1; j < N; j++)

swap(&(a[i\*N+j]), &(a[j\*N + i]));

return a;

}

int\* sw\_sec(int \*a){//работает

int row, col;

for (row = 0; row < N-1; row++ )

for (col = 0; col < N-row-1; col++ )

swap(&(a[row\*N+col]), &(a[(N-col-1)\*N+N-row-1]));

return a;

}

void swap(int \*a, int \*b){//работает

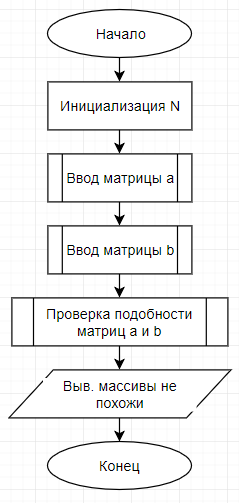
int path = \*a;

\*a = \*b;

\*b = path;

}

void out\_arr(int \*a){//работает

 int i;

for(i = 0; i < N\*N; i++){

printf("%d ", a[i]);

if((i+1)%N == 0)

printf("\n");

}

}

int equal\_arr(int \*a, int \*b){//работает

int i,j;

for(i = 0; i < N; i++)

for(j = 0; j < N; j++)

if(a[i\*N+j] != b[i\*N+j])

return 0;

return 1;

}

int check\_arr(int \*a, int \*b, int t){//

if(equal\_arr(a, b)){

printf("Массивы могут быть похожи");

\_Exit(0);

}

if(t <= 4){

int p1[N\*N], p2[N\*N], p3[N\*N], p4[N\*N];

copy\_arr(a, p1);

copy\_arr(a, p2);

copy\_arr(b, p3);

copy\_arr(b, p4);

check\_arr(b, sw\_main(p1), t+1);

check\_arr(b, sw\_sec(p2), t+1);

check\_arr(sw\_main(p3), a, t+1);

check\_arr(sw\_sec(p4), a, t+1);

}

}

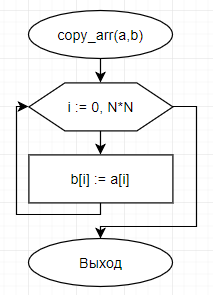
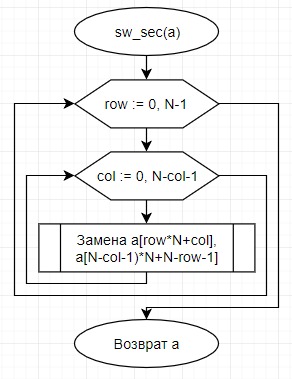
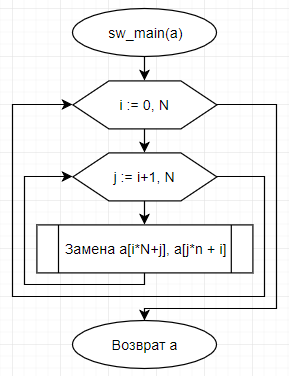
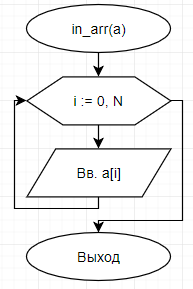
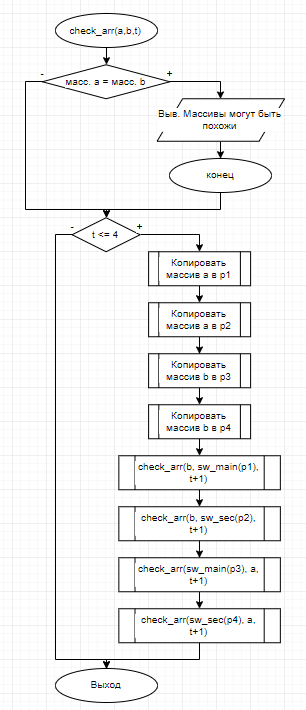
void copy\_arr(int \*a, int \*b){

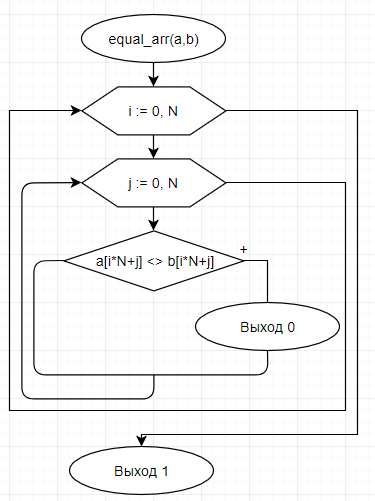
int i;

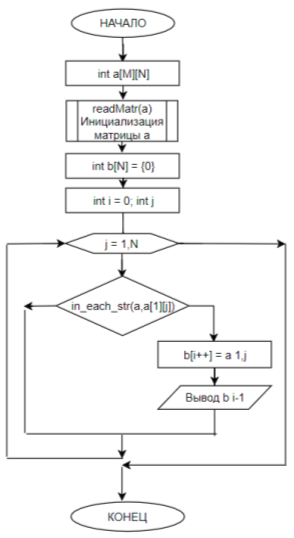
for(i = 0; i < N\*N; i++)

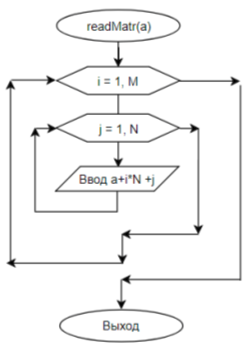
b[i] = a[i];

}





ТЕМА 6. ЛИНЕЙНЫЙ ПОИСК

Найти все числа, каждое из которых встречается в каждой строке матрицы А(m,n).

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <conio.h>

#define M 4 // Количество строк

#define N 4 // Колличество столбцов

void readMatr(int \*\*); // Ввод матрицы

int in\_each\_str(int [][N],int); // Проверка, есть ли наше число в каждой строке матрицы

int in\_str(int [][N],int,int); // Проверка, есть ли число в строке матрицы

int main()

{

setlocale(0,"Rus");

int a[M][N];

readMatr(a);

int b[N] = {0}; //Массив в котором мы сохраним числа, соответствующие условию

int i = 0;

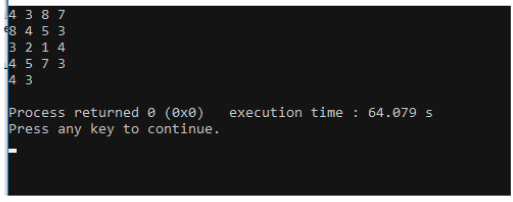
int j;

for (j = 0; j < N;j++){ //Цикл, проверяющий каждое число по условию задачи

if (in\_each\_str(a,a[1][j])){ //Если число подходит, то мы сохраняем его и выводи

b[i++] = a[1][j];

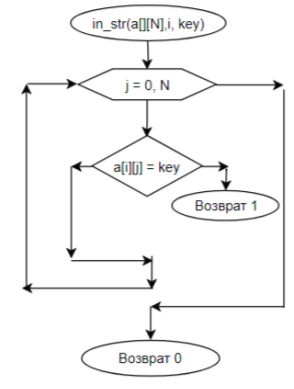
printf("%d ",b[i-1]);

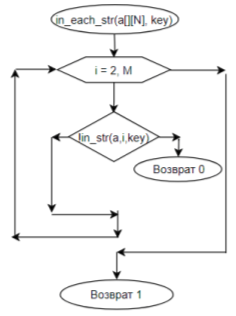
}

}

printf("\n");

return 0;

}

int in\_str(int a[][N], int i,int key){

int j;

for(j = 0; j < N; j++)

if(a[i][j] == key)

return 1;

return 0;

}

int in\_each\_str(int a[][N],int key){

int i;

for(i = 1; i < M; i++)

if (!in\_str(a,i,key))

return 0;

return 1;

}

void readMatr(int \*\*a){

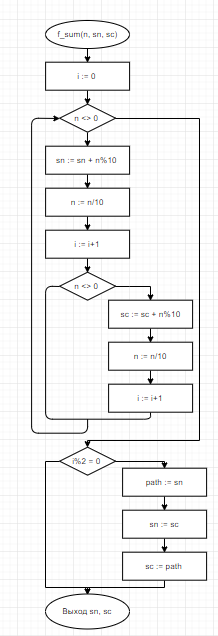
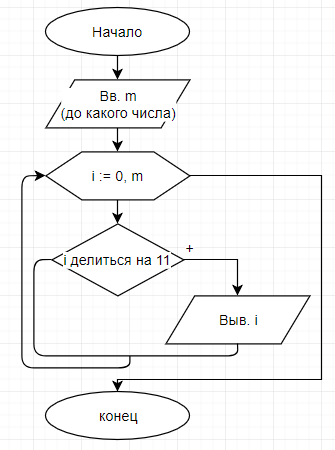
int i,j;

for(i = 0; i < M; i++)

for(j = 0; j < N; j++)

scanf("%d",(a+i\*N+j));

}

ТЕМА 7. АРИФМЕТИКА

Число делится на 11, если разность между суммой цифр, стоящих

на нечётных местах, и суммой цифр, стоящих на чётных местах,

кратна 11. Проверить этот признак для всех натуральных чисел,

не превосходящих заданного , и вывести числа, кратные 11.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

int f\_del11(int);//проверка деления на 11

void f\_sum(int , int\* , int\* );/\*число n, сумма на нечётных sn, на чётных sc\*/

int main()

{

int m;

setlocale(LC\_ALL, "Rus");

printf("Введите до какого числа проверять: ");

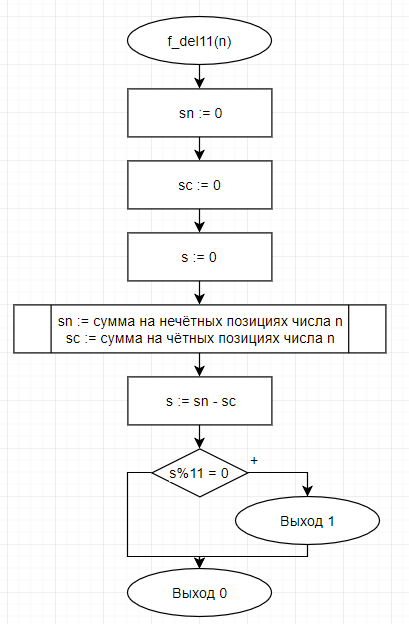
scanf("%d", &m);

int i;

for(i = 0; i <= m; i++)

if(f\_del11(i))

printf("%d ", i);

 return 0;

}

int f\_del11(int n){

int sn = 0,sc = 0;

int s = 0;

f\_sum(n, &sn, &sc);

s = sn - sc;

if((s%11) == 0)

return 1;

return 0;

}

void f\_sum(int n, int \*sn, int \*sc){/\*число n, сумма на нечётных sn, на чётных sc\*/

int i = 0;

while(n != 0){

\*sn = \*sn + n%10;

n = n/10;

i++;

if(n != 0){

\*sc = \*sc + n%10;

n = n/10;

i++;

}

}

if(i%2 == 0){

int path = \*sn;

\*sn = \*sc;

\*sc = path;

}

}

ТЕМА 8. ГЕОМЕТРИЯ И ТЕОРИЯ МНОЖЕСТВ

Из множества n-мерных векторов, заданных своими целочисленными координатами, найти пары ортогональных либо коллинеарных векторов.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <conio.h>

#define N 3 // Размерность векторов

#define COLVEKT 4 // Количество векторов

typedef struct{

int x[N];

} vect;

//Ввод векторов

void read\_vect(vect [COLVEKT]);

//Проверка на ортогональность

int check\_ort(vect, vect);

//Проверка на коллинеарность

int check\_kol(vect, vect);

int main()

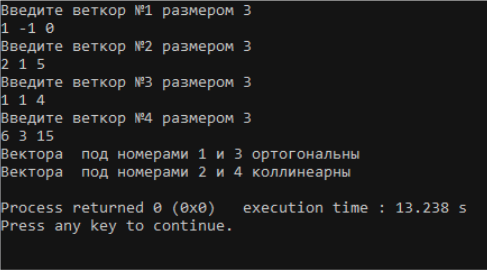
{

setlocale(0,"Rus");

vect a[COLVEKT];

read\_vect(a);

//Проверка каждого вектора с каждым на ортогональность и коллинеарность

int i,j;

for (i = 0; i < COLVEKT; i++){

for (j = i + 1; j < COLVEKT; j++)

if( check\_ort(a[i],a[j]))

printf("Вектора под номерами %d и %d ортогональны\n",i+1,j+1);

if(check\_kol(a[i],a[j]))

printf("Вектора под номерами %d и %d коллинеарны\n",i+1,j+1);

}

return 0;

}

//Проверка на коллинеарность, скал. Произведение равно нулю

int check\_kol(vect a, vect b){

double t = (double) a.x[0] / b.x[0];

double f;

int i;

for (i = 1; i < N; i++){

f = (double) a.x[i] / b.x[i];

if(( !(f - t) < 0.01 ))

return 0;

}

return 1;

}

// Проверка на ортогональность, сохранение пропорции

int check\_ort(vect a, vect b){

int sum = 0;

int i;

for (i = 0; i < N; i++)

sum += a.x[i] \* b.x[i];

if(sum != 0)

return 0;

else

return 1;

}

// Инициализация массива векторов

void read\_vect(vect a[COLVEKT]){

int i,j;

for (i = 1; i <= COLVEKT; i++){

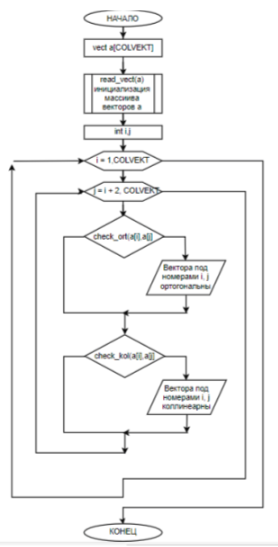
printf("Введите веткор №%d размером %d\n",i,N);

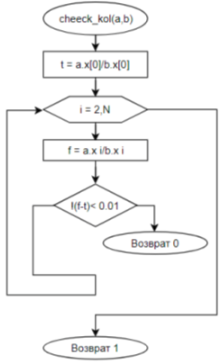
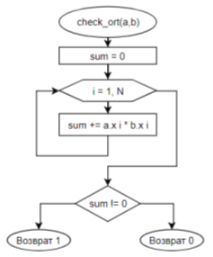
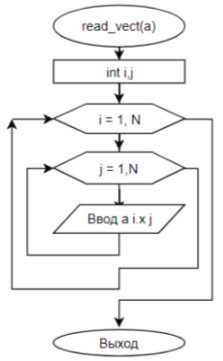
for (j = 0; j < N; j++)

scanf("%d",&a[i - 1].x[j]);

}

}





ТЕМА 9. ЛИНЕЙНАЯ АЛГЕБРА И СЖАТИЕ ИНФОРМАЦИИ

Найти произведение матриц *A(m,n) B(n,k): C(m,k) = A x B.*

Матрицы *A*, *B* и *C* описать как одномерные массивы, используя приведённые индексы.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

//произведение матриц

void in\_arr(int\*, int, int);// Ввод элементов матрицы

void out\_arr(int [], int, int);

void mult\_arr(int\*, int \*, int \*, int, int, int);

int main()

{

setlocale(LC\_ALL, "Rus");

int row1, row2, col1, col2;

//Ввод элементов первой матрицы

printf("Введите количество строк первой матрицы: ");

scanf("%d", &row1);

printf("Введите количество столбцов первой матрицы: ");

scanf("%d", &col1);

printf("---------------------------------------\n");

int a[row1\*col1];

in\_arr(a ,row1, col1);

// Ввод элементов второй матрицы

printf("Введите количество строк второй матрицы: ");

scanf("%d", &row2);

printf("Введите количество столбцов второй матрицы: ");

scanf("%d", &col2);

printf("---------------------------------------\n");

int b[row2][col2];

in\_arr(b, row2, col2);

printf("---------------------------------------\n");

// Вывод элементов второй матрицы

out\_arr(b, row2, col2);

printf("---------------------------------------\n");

if (col1 != row2){

printf("Умножение невозможно!");

return 0;

}

// Умножение матриц

int c[row1\*col2];

mult\_arr(c, a, b, row1, col1, col2);

// Вывод матрицы произведения

printf("Матрица произведения\n");

printf("---------------------------------------\n");

out\_arr(c, row1, col2);

printf("---------------------------------------\n");

return 0;

}

void in\_arr(int \*a, int row, int col){//работает

int i,j;

printf("Введите элементы матрицы\n");

for (i = 0; i < (row\*col); i++)

scanf("%d", &a[i]);

printf("---------------------------------------\n");

}

void out\_arr(int a[], int row, int col){//работает

int i;

for (i = 0; i < row\*col; i++){

printf("%d ", a[i]);

if((i+1)%row == 0)

printf("\n");

}

}

void mult\_arr(int \*c ,int \*a, int \*b, int row1, int col1, int col2){

int i,j,k;

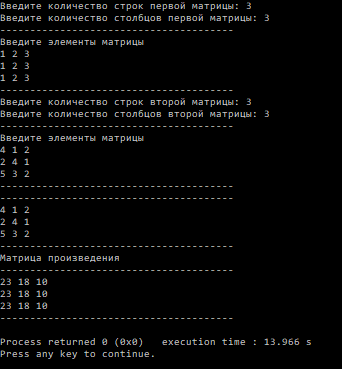
for (i = 0; i < row1; i++){

for (j = 0; j < col2; j++){

c[i\*row1+j] = 0;

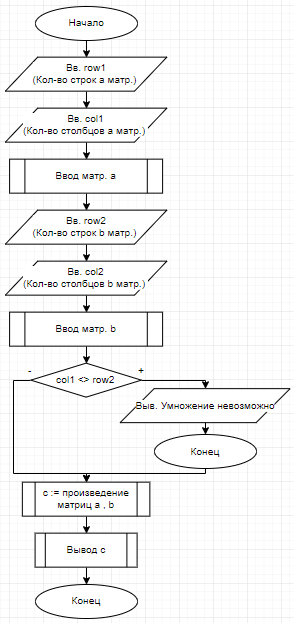
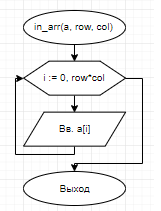
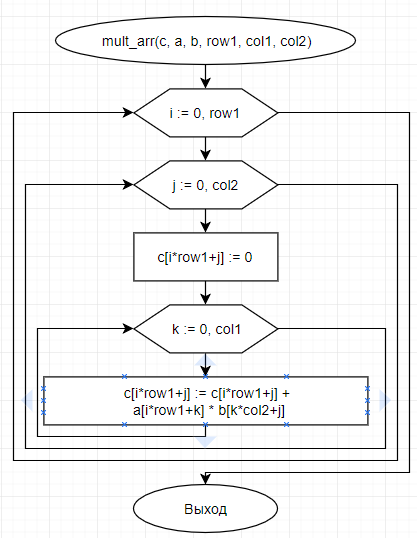
for(k = 0; k < col1; k++)

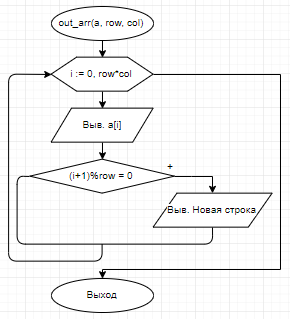
c[i\*row1+j] += a[i\*row1+k] \* b[k\*col2+j];

 }

}

}





ТЕМА 10. АЛГОРИТМЫ ОБРАБОТКИ СИМВОЛЬНОЙ ИНФОРМАЦИИ

Дан текст программы на каком-либо алгоритмическом языке и словарь

зарезервированных слов этого языка (в английской транскрипции).

Преобразовать текст, записав все зарезервированные слова прописными буквами,

а остальные конструкции (имена и так далее) — строчными.

Русские буквы (имена, литералы) не заменять

#include <stdio.h>

#define Write(a,f) fwrite(&a, sizeof(char), 1, f)

#define Read(a,f) fread(&a, sizeof(char), 1, f)

char Read\_word(FILE \*f, char s[], char \*c){

int j=0;

while(Read(\*c,f))

if((\*c >= 'A')&&(\*c <= 'Z')||(\*c >= 'a')&&(\*c <= 'z'))

s[j++] = \*c;

else{ s[j] = '\0';

return \*c; }

}

int include(char nametwo[], char s[]){

FILE \*f = fopen(nametwo,"r");

char c, e[128];

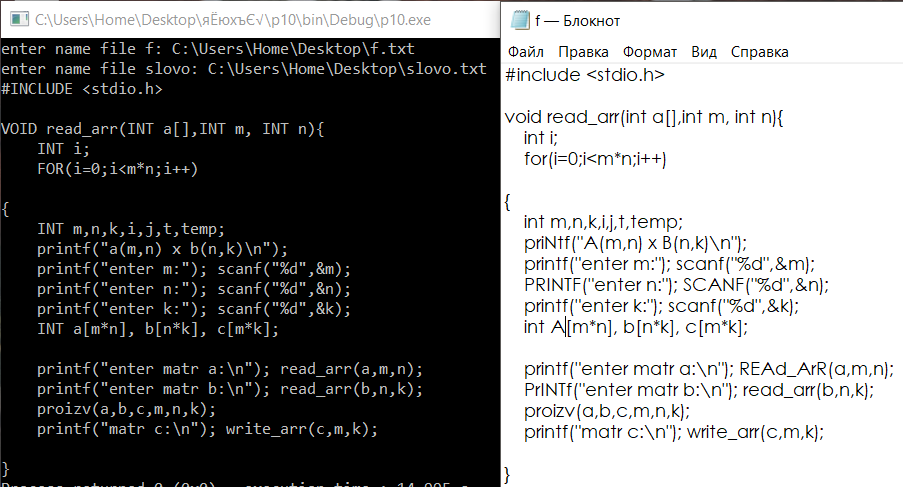
int i;

while(Read\_word(f,e,&c)){

i=0;

while((s[i]==e[i])&&(s[i]!='\0'))

i++;

 if(s[i]==e[i]){

fclose(f);

return 1;

}

}

fclose(f);

return 0;

}

void aAbBcC(char s[]){

int i=0;

while(s[i]!='\0'){

if((s[i]>'a'-1)&&(s[i]<='z'))

s[i] = s[i]-('a'-'A');

i++;

}

}

void AaBbCc(char s[]){

int i=0;

while(s[i]!='\0'){

if((s[i] > 'A'-1)&&(s[i]<='Z'))

s[i] = s[i]+('a'-'A');

i++;

}

}

void main()

{

char nameone[30], nametwo[128], namethree[128], c, s[128];

printf("enter name file f: "); scanf("%s",nameone); // C:\Users\Home\Desktop\f.txt

printf("enter name file slovo: "); scanf("%s",nametwo);

FILE \*f = fopen(nameone,"r");

while(Read\_word(f,s,&c)){

if(include(nametwo,s))

aAbBcC(s);

else AaBbCc(s);

printf("%s",s);

printf("%c",c);

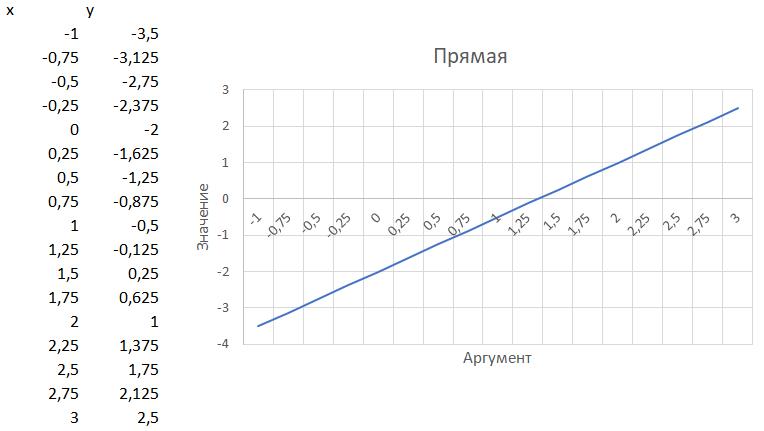
}

fclose(f);

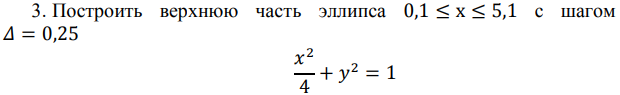
}

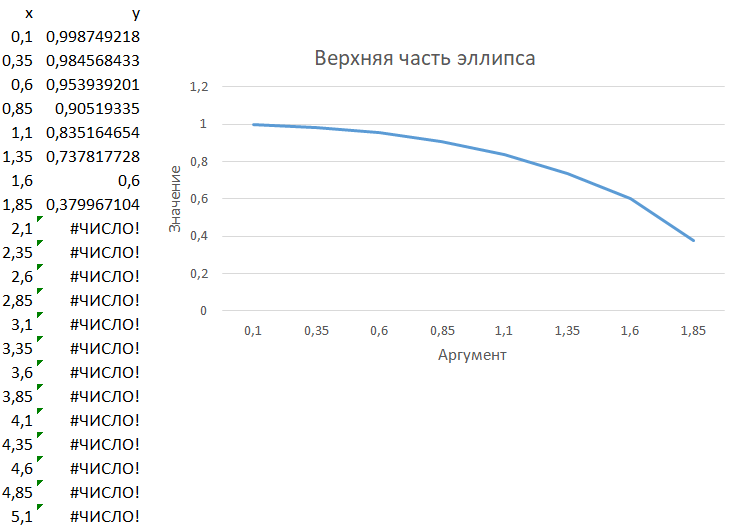
ТЕМА 11. АНАЛИТИЧЕСКАЯ ГЕОМЕТРИЯ



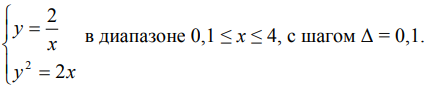


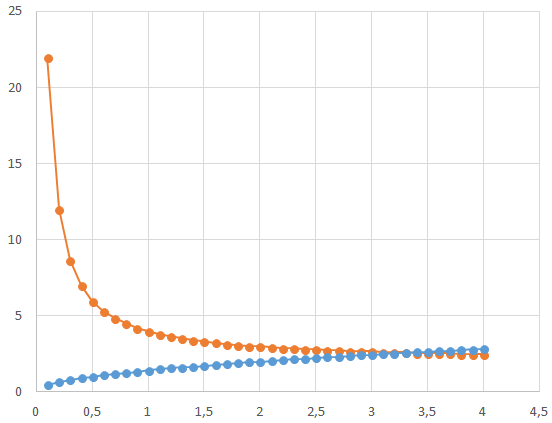
ТЕМА 12. КРИВЫЕ ВТОРОГО ПОРЯДКА НА ПЛОСКОСТИ





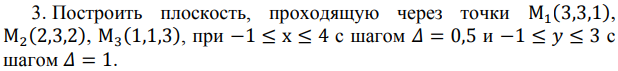
ТЕМА 13. ГРАФИЧЕСКОЕ РЕШЕНИЕ СИСТЕМ УРАВНЕНИЙ

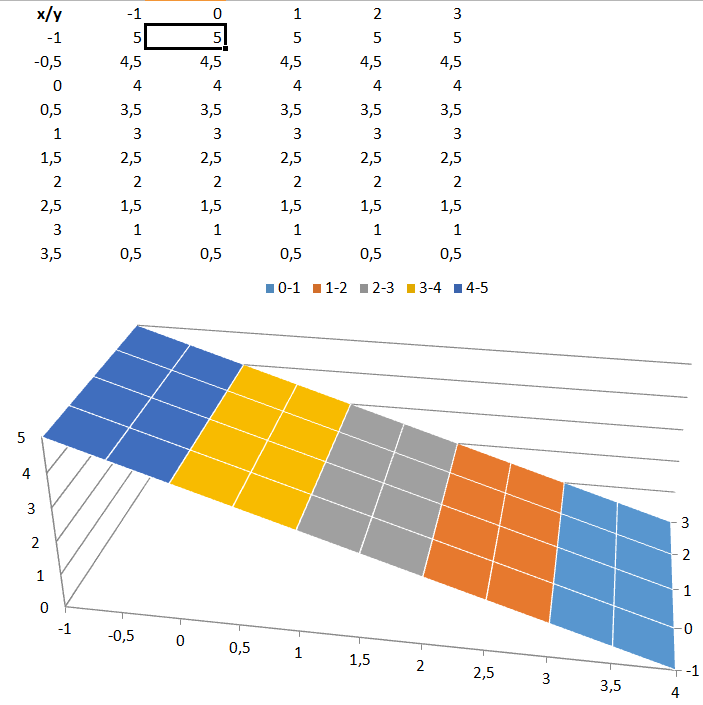




|  |  |  |
| --- | --- | --- |
| x | y1 | y2 |
| 0,1 | 22 | 0,447214 |
| 0,2 | 12 | 0,632456 |
| 0,3 | 8,666667 | 0,774597 |
| 0,4 | 7 | 0,894427 |
| 0,5 | 6 | 1 |
| 0,6 | 5,333333 | 1,095445 |
| 0,7 | 4,857143 | 1,183216 |
| 0,8 | 4,5 | 1,264911 |
| 0,9 | 4,222222 | 1,341641 |
| 1 | 4 | 1,414214 |
| 1,1 | 3,818182 | 1,48324 |
| 1,2 | 3,666667 | 1,549193 |
| 1,3 | 3,538462 | 1,612452 |
| 1,4 | 3,428571 | 1,67332 |
| 1,5 | 3,333333 | 1,732051 |
| 1,6 | 3,25 | 1,788854 |
| 1,7 | 3,176471 | 1,843909 |
| 1,8 | 3,111111 | 1,897367 |
| 1,9 | 3,052632 | 1,949359 |
| 2 | 3 | 2 |
| 2,1 | 2,952381 | 2,04939 |
| 2,2 | 2,909091 | 2,097618 |
| 2,3 | 2,869565 | 2,144761 |
| 2,4 | 2,833333 | 2,19089 |
| 2,5 | 2,8 | 2,236068 |
| 2,6 | 2,769231 | 2,280351 |
| 2,7 | 2,740741 | 2,32379 |
| 2,8 | 2,714286 | 2,366432 |
| 2,9 | 2,689655 | 2,408319 |
| 3 | 2,666667 | 2,44949 |
| 3,1 | 2,645161 | 2,48998 |
| 3,2 | 2,625 | 2,529822 |
| 3,3 | 2,606061 | 2,569047 |
| 3,4 | 2,588235 | 2,607681 |
| 3,5 | 2,571429 | 2,645751 |
| 3,6 | 2,555556 | 2,683282 |
| 3,7 | 2,540541 | 2,720294 |
| 3,8 | 2,526316 | 2,75681 |
| 3,9 | 2,512821 | 2,792848 |
| 4 | 2,5 | 2,828427 |

ТЕМА 14. ПЛОСКОСТЬ В ТРЕХМЕРНОМ ПРОСТРАНСТВЕ





ТЕМА 15. ПОВЕРХНСТЬ ВТОРОГО ПОРЯДКА В ТРЕХ МЕРНОМ ПРОСТРАНСТВЕ

