**Отчёт**

Тема 1. Работа с динамической памятью.

Вариант 19:

**Задание 1**

Заполнить двумерный числовой массив В(3, 5) целыми числами. Найти номера элементов массива больше заданного. Вывести сообщение, если такие элементы не найдены.

**Текст программы:**

#include <iostream>

#include <stdio.h>

#include <locale.h>

#include <cstdlib>

using namespace std;

int main(int argc, const char \* argv[])

{

int mas[3][5], n;

bool isMore = false;

cout << "Заполненный рандомно массив 3Х5:\n";

int count = 0;

for (int i=0; i<3; i++)

{

for (int j=0; j<5; j++)

{

count++;

mas[i][j] = 0 + rand() % 100;

printf("%i: %i\n", count, mas[i][j]);

}

}

cout << "Введите заданное число от 0 до 100: ";

scanf("%i", &n);

printf("Числа больше заданного:\n");

for (int i=0; i<3; i++){

for (int j=0; j<5; j++)

{

if (mas[i][j] > n)

{

isMore = true;

printf("%i\n", mas[i][j]);

}

}

}

if (isMore == false)

{

printf("Больше заданного числа не найдено\n");

}

cin.get();

return 0;

}

**Результат работы:**   
Изображение выглядит как снимок экрана

Автоматически созданное описаниеИзображение выглядит как снимок экрана

Автоматически созданное описание

**Блок схема:**

**Изображение выглядит как текст

Автоматически созданное описание**

**Задание 3(5)**

Скопировать текстовый файл в новый файл, перенести в него только те строки, в которых встречаются буквы от ‘а’ до ‘z”, и указав после каждой строки сумму встретившихся букв (Посчитать сколько всего АНГЛИЙСКИХ МАЛЕНЬКИХ букв от a до z в строке и указать с новой строки, если там только русские, то не выписывать строку)

**Текст программы:**

#include "pch.h"

#include <iostream>

#include<string>

#include<fstream>

using namespace std;

//функция проверяет принадлежность символа русскому алфавиту

bool isRussian(char c)

{

return string("абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ ").find(c) != string::npos;

}

int main()

{

string fin, fon,s;

int k;

bool flr;

setlocale(LC\_ALL, "RUS");

cout << "Введите имя входного файла ";

cin >> fin;

ifstream fi(fin.c\_str());

if (!fi)

{

cout << "Ошибка открытия файла\n"<<"Нажмите Enter...";

getchar();

getchar();

return(-1);

}

cout << "Введите имя выходного файла ";

cin >> fon;

ofstream fo(fon.c\_str());

while (!fi.eof())//пока не кончится файл

{

int k = 0;

fi >> s;//читаем строки

flr = true;//признак того, что строка только из русских букв

for (int i = 0; i < s.length(); i++) //посимвольно рассматриваем строку

{

if ('a' <= s[i] && s[i] <= 'z') k++;

if (!isRussian(s[i])) flr = false;

}

if (!flr) //если строка не только из русских 00

{

fo << s << endl << k<<endl;//выводим строку в файл, а так же количество маленьких

//латинских

// cout << s << endl << k << endl;

}

}

fi.close();

fo.close();

cout << "Нажмите Enter...";

getchar();

getchar();

}

**Результат работы**

****

**Файл 1.txt**

qwerty

йцукен

zxcvb123

ячсмитasd

12345

ваВВВЙЙЙ

**Файл 2.txt**

qwerty

6

zxcvb123

5

ячсмитasd

3

12345

0

**Блок схема:**

fin,fon

ifstream fi(fin.c\_str())

ofstream fo(fon.c\_str())

начало

!fi.eof()

fi >> s

k=0

flr=true

i=0;length(s)-1;1

'a' <= s[i] && s[i] <= 'z'

K++

!isRussian(s[i])

flr=false

!flr

fo << s << endl <<k k<<endl

конец

да

да

да

да

нет

нет

нет

нет

Тема 2. Использование функций.

Вариант 5:

**Задача 2.1 \***

Разработать спецификации и создать функцию для вычисления кратчайшего расстояния от произвольной точки плоскости с координатами X, Y до контура окружности с центром в начале координат и радиусом R. Точка может располагаться внутри или вне фигуры. Организовать вызов функции для проверки.

**Текст программы:**

#include "pch.h"

#include <iostream>

#include<math.h>

double Distance(double X, double Y, double R)

{

return(abs(R-sqrt(X\*X + Y \* Y))); //искомое расстояние равно модулю разности радиуса окружности

//и расстояния от точки до начала координат

}

using namespace std;

int main()

{

double x, y,r;

setlocale(LC\_ALL, "RUS");

cout << "Введите координаты точки\nX=";

cin >> x;

cout << "Y=";

cin >> y;

cout << "Введите радиус окружности\nR=";

cin >> r;

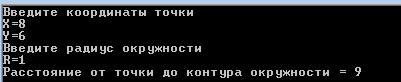
//вызов функции

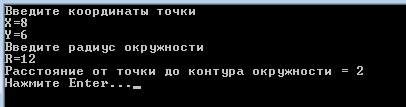
cout << "Расстояние от точки до контура окружности = "<<Distance(x,y,r)<<"\n";

return 0;

}

**Скриншоты работы:**

****

****

**Блок схема:**

Ввод X,Y,R

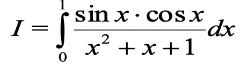
начало

конец

Вывод abs(R-sqrt(X\*X + Y \* Y))

**Задача 2.2**

Разработать программу для вычисления интеграла методом трапеции и методом Симпсона, оформив каждый способ в виде отдельной функции. Вывести на экран результаты интегрирования разными методами для сравнения



**Текст программы:**

#include "pch.h"

#include <iostream>

#include<math.h>

using namespace std;

double f(double x)

{

return(sin(x)\*cos(x) / (x\*x + x + 1));

}

//симпсона

double Simpson(double a, double b, int n)

{

double I = 0,s4=0,s2=0,h;

h = (b - a) / n;//Шаг интегрирования.

for (int i = 1; i <= n - 1; i +=2)

{

s4 += f(a + h \* i);//Значения с нечётными индексами, которые нужно умножить на 4.

s2 += f(a + h \* (i + 1));//Значения с чётными индексами, которые нужно умножить на 2.

}

I = h\*(f(a) + 4 \* s4 + 2 \* s2 - f(b))/3.;//Отнимаем значение f(b) так как ранее прибавили его дважды.

return I;

}

//трапеций

double Trapezium(double a, double b, int n)

{

double I = 0,h;

h = (b - a) / n;//Шаг интегрирования.

for (int i = 0; i < n; i ++)

{

I += (f(a + i \* h) + f(a + (i + 1)\*h))\*h / 2; //суммируем площадь трапеций

}

return I;

}

int main()

{

int n;

setlocale(LC\_ALL, "RUS");

cout << "Введите количество точек разбиения (четное) = ";

cin >> n;

if (n % 2 != 0)

{

cout << "Введено нечетное число\n";

return(-1);

}

cout << "Интеграл методом Симпсона = "<<Simpson(0,1,n);

cout << "\nМетодом трапеций =" << Trapezium(0, 1, n)<<endl;

return(0);

}

**Скриншоты работы**

****

****

**Блок схемы:**

**Метод Симпсона**

a,b,n

h = (b - a) / n

начало

i=1;n-1;2

конец

s4 += f(a + h \* i)

s2 += f(a + h \* (i + 1))

I = h\*(f(a) + 4 \* s4 + 2 \* s2 - f(b))/3

**Метод трапеций**

a,b,n

h = (b - a) / n

начало

i=1;n-1;1

конец

I+= (f(a + i \* h) + f(a + (i + 1)\*h))\*h / 2

Тема 3. Указатели и динамическое распределение памяти

**Задача 3.1**

Динамически выделит память под N элементов вещественного типа float. Разработать алгоритм и программу обработки данных с использованием указателей. Переставить в обратном порядке элементы, расположенные до максимального элемента.

**Текст программы:**

#include "pch.h"

#include <iostream>

using namespace std;

int main()

{

int N;

float \*x,t;

int i, imax;

setlocale(LC\_ALL, "RUS");

cout << "Введите количество элементов N=";

cin >> N;

x = new float[N]; //выделение памяти

cout << "Введите элементы\n";

imax = 0;

for (i = 0; i < N; i++) //цикл ввода элементов массива

{

cout << "x[" << i << "] = ";

cin >> x[i];

if (x[i] > x[imax]) imax = i; //поиск индекса максимального

}

//перестановка в обратном порядке до максимального элемента

for (i = 0; i < imax / 2; i++)

{

t = x[i];

x[i] = x[imax - i - 1];

x[imax - i - 1] = t;

}

//вывод результата

cout << "\nИзмененный массив:\n";

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

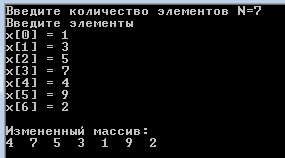
cout << x[i] << " ";

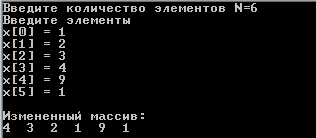
cout << endl;

return(0);

}

**Скриншоты работы:**

****



Ввод n

x = new float[N];

начало

i=0;n-1;1

x[i] > x[imax])

imax=i

конец

да

нет

Ввод x

i=0;imax/2-1;1

t = x[i];

x[i] = x[imax - i - 1];

x[imax - i - 1] = t;

Вывод x

**Задача 3.2\***

Динамически выделить память под N\*M элементов целого типа int. Разработать программу обработки двумерного динамического массива. Найти минимальный и максимальный элемент каждого столбца и поменять их местами с первым и последним элементом столбца соответственно.

**Текст программы:**

#include "pch.h"

#include <iostream>

using namespace std;

int main()

{

int N, M, i, j, imax, imin, t;

setlocale(LC\_ALL, "RUS");

cout << "Введите N = ";

cin >> N;

cout << "Введите M = ";

cin >> M;

//выделение памяти

int \*\*a = new int \*[N];

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

a[i] = new int[M];

cout << "Введите элементы массива\n";

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

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

{

cout << "a["<< i<< "]["<< j<< "] = ";

cin >> a[i][j];

}

//исходный массив

cout << "Исходный массив:\n";

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

{

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

cout << a[i][j]<<"\t";

cout << endl;

}

//поиск минимального в столбцах

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

{

imin = 0;

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

if (a[imin][j] > a[i][j]) imin = i;

//меняем соотв. элементы местами

t = a[0][j];

a[0][j] = a[imin][j];

a[imin][j] = t;

}

//поиск максимального в столбцах

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

{

imax = 0;

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

if (a[imax][j] < a[i][j]) imax = i;

//меняем соотв. элементы местами

t = a[N - 1][j];

a[N - 1][j] = a[imax][j];

a[imax][j] = t;

}

//вывод результата

cout << "Результат:\n";

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

{

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

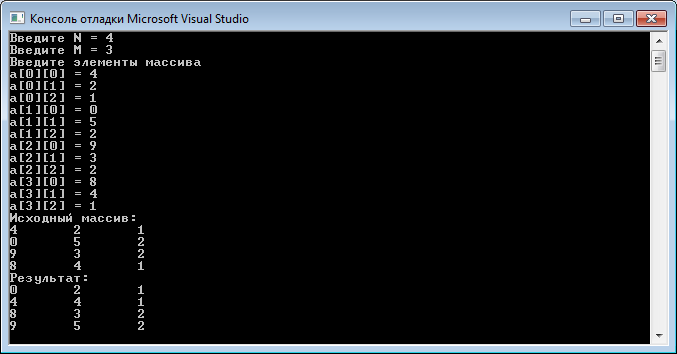
cout << a[i][j]<<"\t";

cout << endl;

}

}

**Скриншот работы**

****

Ввод N,M,a[N][M]

начало

j=0;n-1;1

a[imin][j]>a[i][j]

imin=i

конец

да

нет

t = a[0][j];

a[0][j] = a[imin][j];

a[imin][j] = t;

Вывод a

i=0;n-1;1

j=0;n-1;1

a[imax][j]<a[i][j]

imax=i

да

нет

t = a[N-1][j];

a[N-1][j] = a[imax][j];

a[imax][j] = t;

i=0;n-1;1