\*\*\*\*\*

Реализовать чтение ID3v1 MP3-файла.

a. Название

b. Исполнитель

c. Альбом

d. Год

e. Жанр

#include <iostream>

#include <fstream>

using namespace std;

struct ID3v1Header {

char tag[3];

char title[30];

char artist[30];

char album[30];

char year[4];

char comment[30];

unsigned char genre;

};

bool readID3v1HeaderFromFile(string fileName, ID3v1Header& header)

{

ifstream inFile(fileName, ios::binary);

if (!inFile.is\_open()) {

return false;

}

inFile.seekg(-128, ios::end);

char tag[3];

inFile.read(tag, 3);

if (tag[0] == 'T' && tag[1] == 'A' && tag[2] == 'G') {

inFile.read(header.title, 30);

inFile.read(header.artist, 30);

inFile.read(header.album, 30);

inFile.read(header.year, 4);

inFile.read(header.comment, 30);

inFile.read((char\*)&header.genre, 1);

inFile.close();

return true;

}

inFile.close();

return false;

}

int main()

{

string fileName = "test.mp3";

ID3v1Header header;

if (readID3v1HeaderFromFile(fileName, header)) {

cout << "Title: " << header.title << endl;

cout << "Artist: " << header.artist << endl;

cout << "Album: " << header.album << endl;

cout << "Year: " << header.year << endl;

cout << "Genre: " << (int)header.genre << endl;

}

else {

cout << "ID3v1 header not found in file." << endl;

}

return 0;

}

\*\*\*\*\*

Протабулировать функцию y = sin(x) . Значения аргумента х и функции у на каждом шаге записывать в файл.

#include <iostream>

#include <fstream>

#include <cmath>

using namespace std;

const double PI = 3.14159265358979323846;

int main()

{

ofstream outfile("tablica.txt");

if(!outfile.is\_open())

{

cout << " ошибка!\n";

return 1;

}

double x, step = 0.1, xmin = 0.0, xmax = 2.0 \* PI;

for(x = xmin; x <= xmax; x += step)

{

double y = sin(x);

outfile << x << "\t" << y << endl;

}

outfile.close();

cout << " успешно!\n";

return 0;

}

\*\*\*\*\*

Сгенерировать массив со случайными числами в диапазоне [-100,100] . Записать их в файл.

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() { srand(time(NULL)); // инициализируем генератор случайных чисел текущим временем

const int N = 10; // количество элементов массива

int arr[N];

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

arr[i] = rand() % 201 - 100; // генерируем число в диапазоне [-100,100]

}

ofstream fout("numbers.txt"); // открыть

if (!fout.is\_open()) { // проверить

cout << "error open!" << endl;

return 1;

}

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

fout << arr[i] << " ";

}

fout.close();

cout << "zapisano V numbers.txt" << endl;

return 0;

}

\*\*\*\*\*

Дан текст. Заменить все цифры соответствующими словами.

#include <iostream>

#include <string>

using namespace std;

string numberToWord(char digit)

{

switch (digit) {

case '0': return "zero";

case '1': return "one";

case '2': return "two";

case '3': return "three";

case '4': return "four";

case '5': return "five";

case '6': return "six";

case '7': return "seven";

case '8': return "eight";

case '9': return "nine";

default: return "";

}

}

string replaceNumbersWithWords(string text)

{

string result = "";

for (int i = 0; i < text.length(); i++) {

if (isdigit(text[i])) {

result += numberToWord(text[i]);

}

else {

result += text[i];

}

}

return result;

}

int main()

{

string text = "We are living in the 21st century.";

string result = replaceNumbersWithWords(text);

cout << result << endl;

return 0;

}

\*\*\*\*\*

Дана строка. Если ее длина больше 10, то оставить в строке только первые 6 символов, иначе дополнить строку символами 'o' до длины 12.

#include <iostream>

#include <string>

using namespace std;

string processString(string text)

{

if (text.length() > 10) {

return text.substr(0, 6);

}

else {

int numToAppend = 12 - text.length();

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

text += 'o';

}

return text;

}

}

int main()

{

string text1 = "hello world";

string text2 = "short";

string result1 = processString(text1);

string result2 = processString(text2);

cout << result1 << endl;

cout << result2 << endl;

return 0;

}

\*\*\*\*\*

Составить алгоритм, находящий, является ли частью данного слова слово “раз”. Ответ должен быть “да” или “нет”.

#include <iostream>

#include <string>

using namespace std;

bool checkSubstring(string text)

{

int index = text.find("раз");

if (index != -1) {

return true;

}

else {

return false;

}

}

int main()

{

string text1 = "разум";

string text2 = "умный";

bool result1 = checkSubstring(text1);

bool result2 = checkSubstring(text2);

if (result1) {

cout << "Substring 'raz' is present in text1" << endl;

}

else {

cout << "Substring 'raz' is not present in text1" << endl;

}

if (result2) {

cout << "Substring 'raz' is present in text2" << endl;

}

else {

cout << "Substring 'raz' is not present in text2" << endl;

}

return 0;

}

\*\*\*\*\*

Дана строка. Удалите 6-ый символ в ней.

#include <iostream>

#include <string>

using namespace std;

string removeChar(string text, int pos)

{

return text.substr(0, pos) + text.substr(pos + 1);

}

int main()

{

string text = "hello world";

string result = removeChar(text, 5);

cout << result << endl;

return 0;

}

\*\*\*\*\*

Дана матрица. Элементы первой строки — мощность электромотора, второй строки

— категория мотора (число от 1 до 3). Найти самый мощный мотор категории 3.

#include <iostream>

using namespace std;

const int ROWS = 2;

const int COLS = 5;

int findMostPowerfulMotor(int motors[ROWS][COLS])

{

int maxPower = 0;

for (int j = 0; j < COLS; j++) {

if (motors[1][j] == 3 && motors[0][j] > maxPower) {

maxPower = motors[0][j];

}

}

return maxPower;

}

int main()

{

int motors[ROWS][COLS] = {

{100, 200, 150, 300, 240},

{1, 2, 3, 2, 3}

};

int maxPower = findMostPowerfulMotor(motors);

cout << "Most powerful motor in category 3: " << maxPower << endl;

return 0;

}

\*\*\*\*\*

Объявить, реализовать заполнение данными с клавиатуры ИЛИ из файла и вывести

на экран матрицу, описывающую следующие сущности. Предложите свой вариант.

a. Двумерное изображение. Изображение состоит из пикселей. Каждый пиксель

характеризуется яркостью цветовых каналов: красный, синий, зелёный

#include <iostream>

using namespace std;

const int WIDTH = 3;

const int HEIGHT = 3;

struct Pixel {

int red;

int green;

int blue;

};

void fillPixels(Pixel pixels[][WIDTH])

{

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

for (int j = 0; j < WIDTH; j++) {

cout << "Enter red, green and blue color values for pixel at (" << i << "," << j << "): ";

cin >> pixels[i][j].red >> pixels[i][j].green >> pixels[i][j].blue;

}

}

}

void printPixels(Pixel pixels[][WIDTH])

{

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

for (int j = 0; j < WIDTH; j++) {

cout << "(" << pixels[i][j].red << "," << pixels[i][j].green << "," << pixels[i][j].blue << ") ";

}

cout << endl;

}

}

int main()

{

Pixel pixels[HEIGHT][WIDTH];

fillPixels(pixels);

cout << "Image matrix:" << endl;

printPixels(pixels);

return 0;

}

\*\*\*\*\*

Преобразовать матрицу, умножив элементы каждой строки на значение второго элемента этой строки.

элемента этой строки.

#include <iostream>

using namespace std;

const int ROWS = 3;

const int COLS = 3;

void multiplyRow(int matrix[][COLS], int row)

{

int multiplier = matrix[row][1];

for (int j = 0; j < COLS; j++) {

matrix[row][j] \*= multiplier;

}

}

void printMatrix(int matrix[][COLS])

{

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

for (int j = 0; j < COLS; j++) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

}

int main()

{

int matrix[ROWS][COLS] = {

{1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

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

multiplyRow(matrix, i);

}

cout << "Modified matrix:" << endl;

printMatrix(matrix);

return 0;

}

\*\*\*\*\*

Дана матрица. Составить массив, каждый элемент которого равен максимальному

элементу соответствующей строки матрица

#include <iostream>

using namespace std;

const int ROWS = 3;

const int COLS = 3;

void getMaxValues(int matrix[][COLS], int maxValues[])

{

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

int maxValue = matrix[i][0];

for (int j = 1; j < COLS; j++) {

if (matrix[i][j] > maxValue) {

maxValue = matrix[i][j];

}

}

maxValues[i] = maxValue;

}

}

void printArray(int arr[], int size)

{

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

cout << arr[i] << " ";

}

cout << endl;

}

int main()

{

int matrix[ROWS][COLS] = {

{1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

int maxValues[ROWS];

getMaxValues(matrix, maxValues);

cout << "Max values for each row:" << endl;

printArray(maxValues, ROWS);

return 0;

}

\*\*\*\*\*

В квадратной матрице для каждой строки найти минимальный элемент и

переставить его с элементом, стоящим в этой строке на главной диагонали.

#include <iostream>

using namespace std;

const int SIZE = 3;

void swap(int& a, int& b)

{

int temp = a;

a = b;

b = temp;

}

void findAndSwap(int matrix[][SIZE], int row)

{

int minValue = matrix[row][0];

int minIndex = 0;

for (int j = 1; j < SIZE; j++) {

if (matrix[row][j] < minValue) {

minValue = matrix[row][j];

minIndex = j;

}

}

if (minIndex != row) {

swap(matrix[row][row], matrix[row][minIndex]);

}

}

void printMatrix(int matrix[][SIZE])

{

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

for (int j = 0; j < SIZE; j++) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

}

int main()

{

int matrix[SIZE][SIZE] = {

{9, 8, 7},

{6, 5, 4},

{3, 2, 1}

};

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

findAndSwap(matrix, i);

}

cout << "Modified matrix:" << endl;

printMatrix(matrix);

return 0;

}

\*\*\*\*\*

Дана матрица. Элементы первой строки — количество осадков в соответствующий

день, второй строки — сила ветра в этот день. Найти, был ли в эти дни ураган?

(ураган — когда самый сильный ветер был в самый дождливый день)

#include <iostream>

using namespace std;

const int N = 7; // количество дней

int main() {

int precipitations[N];

int winds[N];

int max\_wind = 0;

int max\_wind\_index = 0;

// Вводим данные о погоде за неделю

cout << "Введите количество осадков и силу ветра за каждый день недели: " << endl;

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

cin >> precipitations[i] >> winds[i];

// Находим индекс наибольшего значения второй строки

if (winds[i] > max\_wind) {

max\_wind = winds[i];

max\_wind\_index = i;

}

}

// Проверяем, был ли ураган в этот день

if (precipitations[max\_wind\_index] == \*max\_element(precipitations, precipitations + N)) {

cout << "Был ураган в день " << max\_wind\_index + 1 << endl;

}

else {

cout << "Урагана не было." << endl;

}

return 0;

}

\*\*\*\*\*

В квадратной матрице найти сумму положительных элементов, лежащих на и выше

главной диагонали и расположенных в чётных столбцах.

#include <iostream>

using namespace std;

const int SIZE = 3;

int findSum(int matrix[][SIZE])

{

int sum = 0;

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

for (int j = i; j < SIZE; j++) {

if (j % 2 == 0 && matrix[i][j] > 0) {

sum += matrix[i][j];

}

}

}

return sum;

}

int main()

{

int matrix[SIZE][SIZE] = {

{1, 2, 3},

{4, -5, 6},

{7, 8, -9}

};

int sum = findSum(matrix);

cout << "Sum: " << sum << endl;

return 0;

}

\*\*\*\*\*

Преобразовать матрицу: элементы строки, в которой находится минимальный

элемент матрицы, заменить нулями.

#include <iostream>

using namespace std;

const int N = 3, M = 4;

int main() {

int matrix[N][M] = { {1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12} };

int minElement = matrix[0][0], minRow = 0;

// находим минимальный элемент и его строку

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

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

if (matrix[i][j] < minElement) {

minElement = matrix[i][j];

minRow = i;

}

}

}

// заменяем элементы строки на 0

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

matrix[minRow][j] = 0;

}

// выводим преобразованную матрицу

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

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

cout << matrix[i][j] << " ";

}

cout << endl;

}

return 0;

}

\*\*\*\*\*

По введенному номеру дня недели вывести его название, используя оператор множественного выбора. Нумерация дней недели начинается с 1 – понедельник, 2 — вторник и т. д.

#include <iostream>

using namespace std;

int main() {

int day;

cout << "Введите номер дня недели (от 1 до 7): ";

cin >> day;

switch (day) {

case 1:

cout << "Понедельник" << endl;

break;

case 2:

cout << "Вторник" << endl;

break;

case 3:

cout << "Среда" << endl;

break;

case 4:

cout << "Четверг" << endl;

break;

case 5:

cout << "Пятница" << endl;

break;

case 6:

cout << "Суббота" << endl;

break;

case 7:

cout << "Воскресенье" << endl;

break;

default:

cout << "Некорректный номер дня недели" << endl;

}

return 0;

}

\*\*\*\*\*

Составить алгоритм, исключающий из строки А символы строки Б.

#include <iostream>

#include <string>

using namespace std;

int main() {

string strA, strB;

cout << "Введите строку А: ";

getline(cin, strA);

cout << "Введите строку Б: ";

getline(cin, strB);

bool found;

for (char ch : strA) {

found = false;

for (char c : strB) {

if (ch == c) {

found = true;

break;

}

}

if (!found) {

cout << ch;

}

}

cout << endl;

if (found) {

cout << "Некоторые символы были исключены из строки А" << endl;

} else {

cout << "Все символы строки А были исключены" << endl;

}

return 0;

}

\*\*\*\*\*

В данной строке вставить после каждого символа 'a' символ 'b'.

#include <iostream>

#include <string>

using namespace std;

int main() {

string str, newStr;

cout << "Введите строку: ";

getline(cin, str);

for (char ch : str) {

if (ch == 'a') {

newStr += "ab";

} else {

newStr += ch;

}

}

cout << "Новая строка: " << newStr << endl;

return 0;

}

\*\*\*\*\*

Дана строка. Вставить после каждого символа пробел. Пробел не считается символом, после которого надо ставить пробел.

#include <iostream>

#include <string>

using namespace std;

int main() {

string str, newStr;

cout << "Введите строку: ";

getline(cin, str);

for (int i = 0; i < str.length(); i++) {

if (str[i] != ' ' && i > 0 && str[i-1] != ' ') {

newStr += ' ';

}

newStr += str[i];

}

cout << "Новая строка: " << newStr << endl;

return 0;

}

\*\*\*\*\*

Исключить из строки группы символов, расположенные между символами «/\*» и «\*/» включая границы. Предполагается, что нет вложенных скобок.

#include <iostream>

#include <string>

using namespace std;

int main() {

string str, newStr;

bool excludeFlag = false;

cout << "Введите строку: ";

getline(cin, str);

for (int i = 0; i < str.length(); i++) {

if (i < str.length() - 1 && str[i] == '/' && str[i+1] == '\*') {

excludeFlag = true;

i++;

} else if (i < str.length() - 1 && str[i] == '\*' && str[i+1] == '/') {

excludeFlag = false;

i++;

} else if (!excludeFlag) {

newStr += str[i];

}

}

cout << "Новая строка: " << newStr << endl;

return 0;

}

\*\*\*\*\*

Составить алгоритм, находящий, сколько раз в тексте сочетание “мир” является отдельным словом (пробелы перед и после).

#include <iostream>

#include <sstream>

using namespace std;

bool isSeparateWord(const string& text, int startIndex, int endIndex) {

if (startIndex == 0 && endIndex < text.length() - 1) {

return text[endIndex + 1] == ' ';

}

if (startIndex > 0 && endIndex == text.length() - 1) {

return text[startIndex - 1] == ' ';

}

if (startIndex == 0 && endIndex == text.length() - 1) {

return true;

}

return (text[endIndex + 1] == ' ' && text[startIndex - 1] == ' ');

}

int main() {

string text, word;

int count = 0;

cout << "Введите текст: ";

getline(cin, text);

stringstream ss(text);

while (ss >> word) {

if (word == "мир" && isSeparateWord(text, ss.tellg() - word.length() - 1, ss.tellg() - 1)) {

count++;

}

}

cout << "Количество отдельных слов 'мир': " << count << endl;

return 0;

}

\*\*\*\*\*

Составить алгоритм, находящий, сколько раз в тексте сочетание “ум” встречается не в начале слова.

#include <iostream>

using namespace std;

bool isBeginningOfWord(const string& text, int index) {

if (index == 0) {

return true;

}

return text[index - 1] == ' ';

}

int main() {

string text;

int count = 0;

cout << "Введите текст: ";

getline(cin, text);

for (int i = 0; i < text.length(); i++) {

if (text[i] == 'у' && i < text.length() - 1 && text[i + 1] == 'м' && !isBeginningOfWord(text, i)) {

count++;

}

}

cout << "Количество вхождений 'ум', не являющихся началом слова: " << count << endl;

return 0;

}

\*\*\*\*\*

Составить алгоритм, находящий, сколько слов в тексте состоит из 6 символов.

#include <iostream>

#include <sstream>

using namespace std;

int main() {

string text, word;

int count = 0;

cout << "Введите текст: ";

getline(cin, text);

stringstream ss(text);

while (ss >> word) {

if (word.length() == 6) {

count++;

}

}

cout << "Количество слов, состоящих из 6 символов: " << count << endl;

return 0;

}

\*\*\*\*\*

Составить алгоритм, подсчитывающий количество слов в тексте.

#include <iostream>

using namespace std;

int main() {

string text;

int wordCount = 0;

cout << "Введите текст: ";

getline(cin, text);

for (int i = 0; i < text.length(); i++) {

if (text[i] == ' ') {

wordCount++;

}

}

wordCount++; // учесть последнее слово без завершающего пробела

cout << "Количество слов: " << wordCount << endl;

return 0;

}

\*\*\*\*\*

Удалить из строки первые буквы слов. (Пример. Входная строка: Crystal Method. Результат: rystal ethod).

#include <iostream>

using namespace std;

bool isWordStart(char prevChar, char currentChar) {

return (prevChar == ' ' || prevChar == '-' || prevChar == '\n') && currentChar != ' ';

}

int main() {

string text, result;

cout << "Введите строку: ";

getline(cin, text);

char prevChar = '\n'; // начальное значение предыдущего символа - перевод строки

for (int i = 0; i < text.length(); i++) {

if (isWordStart(prevChar, text[i])) {

i++; // пропустить первую букву слова

while (i < text.length() && text[i] != ' ' && text[i] != '-') {

result += text[i];

i++;

}

result += ' ';

}

prevChar = text[i];

}

cout << result << endl;

return 0;

}

\*\*\*\*\*

Найти самое часто встречающееся слово в тексте.

#include <iostream>

#include <unordered\_map>

#include <sstream>

#include <algorithm>

using namespace std;

int main() {

string text;

cout << "Введите текст: ";

getline(cin, text);

// Инициализация словаря для подсчета количества вхождений слов

unordered\_map<string, int> wordCounts;

// Разбиение строки на слова

istringstream iss(text);

string word;

while (iss >> word) {

// Проверка наличия слова в словаре и увеличение счетчика вхождений

if (wordCounts.find(word) != wordCounts.end()) {

wordCounts[word]++;

} else {

wordCounts[word] = 1;

}

}

// Поиск самого часто встречающегося слова

int maxCount = 0;

string maxWord;

for (auto const& pair: wordCounts) {

if (pair.second > maxCount) {

maxCount = pair.second;

maxWord = pair.first;

}

}

// Вывод результата

cout << "Самое частое слово: " << maxWord << endl;

cout << "Количество вхождений: " << maxCount << endl;

return 0;

}

\*\*\*\*\*

Дана строка. Если она начинается на 'abc', то заменить их на 'www', иначе добавить в конец строки 'zzz'.

#include <iostream>

#include <string>

int main()

{

std::string str;

std::cout << "Введите строку: ";

std::getline(std::cin, str);

if (str.substr(0, 3) == "abc") {

str.replace(0, 3, "www");

} else {

str += "zzz";

}

std::cout << "Результат: " << str << std::endl;

return 0;

}

\*\*\*\*\*

Дан массив строк. Упорядочить массив по длине строк.

#include <iostream>

#include <algorithm>

#include <vector>

#include <string>

int main()

{

std::vector<std::string> strings = {"abc", "a", "abcdefg", "ab", "abcd", "abcdef"};

// Сортируем массив по длине строк

std::sort(strings.begin(), strings.end(),

[](std::string s1, std::string s2) { return s1.size() < s2.size(); });

// Выводим результат

for (const auto& str : strings) {

std::cout << str << std::endl;

}

return 0;

}

\*\*\*\*\*

Составить алгоритм, заменяющий в тексте слова “правда” на “ложь”.

#include <iostream>

#include <string>

#include <algorithm>

int main()

{

std::string text = "Правда и только правда!";

// Заменяем слово "правда" на "ложь"

std::string from = "правда";

std::string to = "ложь";

size\_t start\_pos = 0;

while ((start\_pos = text.find(from, start\_pos)) != std::string::npos) {

text.replace(start\_pos, from.length(), to);

start\_pos += to.length();

}

// Выводим полученный текст

std::cout << text << std::endl;

return 0;

}

\*\*\*\*\*

Реализовать чтение заголовков файлов форматов.(ICO: ICONDIR и ICONDIRENTRY)

#include <iostream>

#include <fstream>

using namespace std;

#pragma pack(push, 2) // устанавливаем выравнивание в 2 байта, чтобы убрать лишние пробелы в структурах

// структура ICONDIR

struct ICONDIR {

WORD idReserved; // Reserved (must be 0)

WORD idType; // Resource Type (1 for icons)

WORD idCount; // How many images?

};

// структура ICONDIRENTRY

struct ICONDIRENTRY {

BYTE bWidth; // Width, in pixels, of the image

BYTE bHeight; // Height, in pixels, of the image

BYTE bColorCount; // Number of colors in image (0 if >=8bpp)

BYTE bReserved; // Reserved (must be 0)

WORD wPlanes; // Color Planes

WORD wBitCount; // Bits per pixel

DWORD dwBytesInRes; // How many bytes in this resource?

DWORD dwImageOffset; // Where in the file is this image?

};

#pragma pack(pop) // возвращаем стандартное выравнивание

int main() {

ifstream file("file.ico", ios::binary); // открываем файл в двоичном режиме

if (!file.is\_open()) { // проверяем, открылся ли файл

cout << "Ошибка открытия файла\n";

return 1;

}

ICONDIR icondir;

file.read((char\*)&icondir, sizeof(ICONDIR)); // читаем ICONDIR

cout << "idReserved: " << icondir.idReserved << endl;

cout << "idType: " << icondir.idType << endl;

cout << "idCount: " << icondir.idCount << endl;

for (int i = 0; i < icondir.idCount; i++) {

ICONDIRENTRY icodirentry;

file.read((char\*)&icodirentry, sizeof(ICONDIRENTRY)); // читаем ICONDIRENTRY

cout << "bWidth: " << (int)icodirentry.bWidth << endl;

cout << "bHeight: " << (int)icodirentry.bHeight << endl;

cout << "bColorCount: " << (int)icodirentry.bColorCount << endl;

cout << "bReserved: " << (int)icodirentry.bReserved << endl;

cout << "wPlanes: " << icodirentry.wPlanes << endl;

cout << "wBitCount: " << icodirentry.wBitCount << endl;

cout << "dwBytesInRes: " << icodirentry.dwBytesInRes << endl;

cout << "dwImageOffset: " << icodirentry.dwImageOffset << endl;

}

file.close(); // закрываем файл

return 0;

}

\*\*\*\*\*

Разработать программу для ввода с клавиатуры и записи в файл полётных данных о

квадрокоптерах. Одна строка содержит: Х,У,Команда, где ХУ – координаты, а

Команда – это набор чисел, характеризующих действие коптера: 10 – взлёт, 20 –

посадка, 50 – полёт, 60 – набрать высоту, 70 снизить высоту. Пользователь вводит

данные с клавиатуры. Каждая новая строка содержит новый набор координат и

команд. Организовать переменные для хранения информации.

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

struct FlightData {

float x;

float y;

int command;

};

bool writeFlightDataToFile(string fileName, FlightData flightData)

{

ofstream outFile(fileName, ios::app);

if (!outFile.is\_open()) {

return false;

}

outFile << flightData.x << "," << flightData.y << "," << flightData.command << endl;

outFile.close();

return true;

}

int main()

{

string fileName = "flight\_data.txt";

FlightData flightData;

while (true) {

cout << "Enter coordinates (x,y): ";

cin >> flightData.x >> flightData.y;

cout << "Enter command (10-takeoff, 20-landing, 50-flight, 60-increase altitude, 70-decrease altitude): ";

cin >> flightData.command;

if (!writeFlightDataToFile(fileName, flightData)) {

cout << "Failed to write flight data to file." << endl;

break;

}

cout << "Flight data has been successfully recorded to file." << endl;

}

return 0;

}

\*\*\*\*\*

Разработать программу для считывания полётных данных о квадрокоптерах из

файла. Одна строка содержит: Х,У,Команда, где ХУ – координаты, а Команда – это

набор чисел, характеризующих действие коптера: 10 – взлёт, 20 – посадка, 50 –

полёт, 60 – набрать высоту, 70 снизить высоту. Каждая новая строка содержит

новый набор координат и команд. Организовать переменные для хранения

информации

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

int main() {

ifstream inp\_file("INPUT.TXT"); // открытие файла на чтение

ofstream out\_file("OUTPUT.TXT"); // открытие файла на запись

int count = 0; // счётчик количества считанных строк

if (inp\_file.is\_open() && out\_file.is\_open()) { // проверка, что оба файла открыты

string line;

while (getline(inp\_file, line)) { // считывание строк из файла

int commands[10]; // массив для хранения команд

int x, y;

char\* str = new char[line.size() + 1];

strcpy(str, line.c\_str()); // преобразование строки в массив char

char\* token = strtok(str, " "); // разделение строки на подстроки по пробелу

x = atoi(token);

token = strtok(NULL, " ");

y = atoi(token);

int i = 0;

while (token != NULL) {

token = strtok(NULL, " ");

if (token != NULL) {

commands[i] = atoi(token);

i++;

}

}

delete[] str; // очистка памяти

count++; // увеличение счётчика

// вывод данных на экран и запись в файл

out\_file << "Координаты: (" << x << ", " << y << ")" << endl;

out\_file << "Действия: ";

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

if (commands[j] == 10) {

out\_file << "взлёт, ";

} else if (commands[j] == 20) {

out\_file << "посадка, ";

} else if (commands[j] == 50) {

out\_file << "полёт, ";

} else if (commands[j] == 60) {

out\_file << "набрать высоту, ";

} else if (commands[j] == 70) {

out\_file << "снизить высоту, ";

}

}

out\_file << endl << endl;

cout << "Координаты: (" << x << ", " << y << ")" << endl;

cout << "Действия: ";

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

if (commands[j] == 10) {

cout << "взлёт, ";

} else if (commands[j] == 20) {

cout << "посадка, ";

} else if (commands[j] == 50) {

cout << "полёт, ";

} else if (commands[j] == 60) {

cout << "набрать высоту, ";

} else if (commands[j] == 70) {

cout << "снизить высоту, ";

}

}

cout << endl << endl;

}

inp\_file.close(); // закрытие файла на чтение

out\_file.close(); // закрытие файла на запись

cout << "Количество считанных строк: " << count << endl; // вывод количества считанных строк на экран

} else {

cout << "Ошибка открытия файлов" << endl; // вывод сообщения об ошибке на экран

}

return 0;

}

\*\*\*\*\*

Пользователь вводит данные о студентах: Фамилия, Средняя оценка, Курс.

Организовать запись данных из массива структур в файл в следующем формате: 1

строка Фамилия, 2 строка Средняя оценка, 3 строка Курс. Остальные строки

повторяют тройками эти данные для других студентов. Организовать переменные

для хранения информации.

#include <iostream> #include <fstream> #include <string>

using namespace std;

struct Student { string surname; double averageMark; int course; };

const int MAX\_STUDENTS = 10;

int main() { Student students[MAX\_STUDENTS]; int numStudents;

cout << "Enter the number of students (up to " << MAX\_STUDENTS << "): ";

cin >> numStudents;

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

cout << "Enter data for student " << i + 1 << ":" << endl;

cout << "Surname: ";

cin >> students[i].surname;

cout << "Average mark: ";

cin >> students[i].averageMark;

cout << "Course: ";

cin >> students[i].course;

}

ofstream outputFile("students.txt");

if (!outputFile.is\_open()) {

cerr << "Error opening file" << endl;

return 1;

}

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

outputFile << "Surname: " << students[i].surname << endl;

outputFile << "Average mark: " << students[i].averageMark << endl;

outputFile << "Course: " << students[i].course << endl;

}

outputFile.close();

return 0;

}

\*\*\*\*\*

Разработать программу для считывания данных о студентах из файла в следующем

формате: 1 строка Фамилия, 2 строка Средняя оценка, 3 строка Курс. Остальные

строки повторяют тройками эти данные для других студентов. Организовать

считывание данных из файла в массив структур. Организовать переменные для

хранения информации.

#include <iostream> #include <fstream> #include <string>

using namespace std;

struct Student { string surname; double averageMark; int course; };

const int MAX\_STUDENTS = 10;

int main() { Student students[MAX\_STUDENTS]; int numStudents = 0;

ifstream inputFile("students.txt");

if (!inputFile.is\_open()) {

cerr << "Error opening file" << endl;

return 1;

}

while (!inputFile.eof() && numStudents < MAX\_STUDENTS) {

inputFile >> students[numStudents].surname;

inputFile >> students[numStudents].averageMark;

inputFile >> students[numStudents].course;

numStudents++;

}

inputFile.close();

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

cout << "Student " << i + 1 << ":" << endl;

cout << "Surname: " << students[i].surname << endl;

cout << "Average mark: " << students[i].averageMark << endl;

cout << "Course: " << students[i].course << endl;

}

return 0;

}

\*\*\*\*\*

Объявить указатели на две переменные указанного типа. Используя указатели

произвести операции сложения и вычитания над переменными. Вывести адреса

указателей. (FLOAT)

#include <iostream>

using namespace std;

int main() {

float a = 3.14, b = 2.71;

float \*ptr1 = &a, \*ptr2 = &b;

float sum = \*ptr1 + \*ptr2;

float diff = \*ptr1 - \*ptr2;

cout << "Address of ptr1: " << ptr1 << endl;

cout << "Address of ptr2: " << ptr2 << endl;

cout << "Sum of a and b: " << sum << endl;

cout << "Difference of a and b: " << diff << endl;

return 0;

}

\*\*\*\*\*

Объявить указатели на две переменные указанного типа. Используя указатели

произвести операции сложения и вычитания над переменными. Вывести адреса

указателей. (INT)

#include <iostream>

using namespace std;

int main() {

int a = 10, b = 5;

int \*ptr1 = &a, \*ptr2 = &b;

int sum = \*ptr1 + \*ptr2;

int diff = \*ptr1 - \*ptr2;

cout << "Address of ptr1: " << ptr1 << endl;

cout << "Address of ptr2: " << ptr2 << endl;

cout << "Sum of a and b: " << sum << endl;

cout << "Difference of a and b: " << diff << endl;

return 0;

}

\*\*\*\*\*

Объявить сущность и описать её свойства. Объявить динамический массив

сущностей. Реализовать функцию, которая изменяет значения элемента (структура)

массива. Заполнить атрибуты числовыми и текстовыми значениями (диапазон

значений определить самостоятельно)

#include<iostream>

using namespace std;

struct Computer {

string name;

int memory;

int disk\_space;

string processor;

};

void change(Computer\* comp\_arr, int index, string name, int memory, int disk\_space, string processor) {

comp\_arr[index].name = name;

comp\_arr[index].memory = memory;

comp\_arr[index].disk\_space = disk\_space;

comp\_arr[index].processor = processor;

}

int main() {

Computer\* comp\_array = new Computer[3];

comp\_array[0].name = "Lenovo";

comp\_array[0].memory = 8;

comp\_array[0].disk\_space = 512;

comp\_array[0].processor = "Intel Core i5";

comp\_array[1].name = "HP";

comp\_array[1].memory = 16;

comp\_array[1].disk\_space = 1024;

comp\_array[1].processor = "Intel Core i7";

comp\_array[2].name = "Dell";

comp\_array[2].memory = 32;

comp\_array[2].disk\_space = 2048;

comp\_array[2].processor = "Intel Xeon";

cout << "Before change:" << endl;

cout << "Name: " << comp\_array[1].name << ", Memory: " << comp\_array[1].memory

<< ", Disk Space: " << comp\_array[1].disk\_space << ", Processor: " << comp\_array[1].processor << endl;

change(comp\_array, 1, "Acer", 12, 750, "AMD Ryzen 7");

cout << "After change:" << endl;

cout << "Name: " << comp\_array[1].name << ", Memory: " << comp\_array[1].memory

<< ", Disk Space: " << comp\_array[1].disk\_space << ", Processor: " << comp\_array[1].processor << endl;

delete[] comp\_array;

return 0;

}

\*\*\*\*\*

Объявить динамический массив. Размер массива задаёт пользователь. Заполнить

массив случайными числами. Вывести на экран адреса и значения элементов

массива.

#include <iostream>

#include <ctime>

#include <cstdlib>

using namespace std;

int main() {

int n; // Размер массива

cout << "Enter size of array: ";

cin >> n;

int\* arr = new int[n];

srand(time(NULL));

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

arr[i] = rand() % 100;

}

cout << "Array elements: " << endl;

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

cout << "Address: " << &arr[i] << ", Value: " << arr[i] << endl;

}

delete[] arr;

return 0;

}

\*\*\*\*\*

Написать функцию для обмена значений переменных, указанных выше. Передача по

указателю. Выполнить программу по шагам и выписать в тетрадь адреса указателей

и величины переменных.

#include <iostream>

using namespace std;

void swap(int\* a, int\* b) { // Функция для обмена значений по указателю

int tmp = \*a;

\*a = \*b;

\*b = tmp;

}

int main() {

int x = 5, y = 10;

cout << "Before swap - x: " << x << ", y: " << y << endl;

swap(&x, &y); // Вызов функции swap с передачей указателей на переменные x и y

cout << "After swap - x: " << x << ", y: " << y << endl;

return 0;

}

\*\*\*\*\*

Реализовать считывание Ширины (bcWidth) и высоты (bcHeight) растра в пикселях из структуры BITMAPINFO заданного BMP-файла.

#include <iostream>

#include <windows.h>

using namespace std;

int main()

{

BITMAPINFO bmpInfo;

memset(&bmpInfo, 0, sizeof(BITMAPINFO)); // обнуляем структуру

// открываем BMP-файл

HANDLE hFile = CreateFile(L"image.bmp", GENERIC\_READ, FILE\_SHARE\_READ, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

if(hFile == INVALID\_HANDLE\_VALUE)

{

cerr << "Ошибка открытия файла!\n";

return 1;

}

// считываем структуру BITMAPINFO из файла

DWORD dwSize;

if(!ReadFile(hFile, &bmpInfo, sizeof(BITMAPINFO), &dwSize, NULL) || dwSize != sizeof(BITMAPINFO))

{

cerr << "Ошибка чтения структуры файла!\n";

CloseHandle(hFile);

return 1;

}

// из структуры BITMAPINFO извлекаем размеры растра

int bcWidth = bmpInfo.bmiHeader.biWidth; // ширина в пикселях

int bcHeight = bmpInfo.bmiHeader.biHeight; // высота в пикселях

cout << "Ширина: " << bcWidth << " пикселей\n";

cout << "Высота: " << bcHeight << " пикселей\n";

CloseHandle(hFile);

return 0;

}

\*\*\*\*\*

В данной строке найти количество цифр

#include <iostream>

#include <cstring>

using namespace std;

int main() {

char str[] = "String with 123 numbers"; // заданная строка

int digitCount = 0; // число цифр в строке

int len = strlen(str); // длина строки

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

if (isdigit(str[i]))

digitCount++;

}

cout << "Number of digits in the string: " << digitCount << endl;

return 0;

}

\*\*\*\*\*

В квадратной матрице, найти сумму отрицательных элементов, лежащих на главной

диагонали

#include <iostream>

#include <ctime>

using namespace std;

const int N = 5; // размер матрицы

int main() {

int matrix[N][N];

int sum = 0;

// Заполняем матрицу случайными числами

srand(time(NULL));

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

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

matrix[i][j] = rand() % 21 - 10; // от -10 до 10

}

}

// Выводим матрицу на экран

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

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

cout << matrix[i][j] << ' ';

}

cout << endl;

}

// Вычисляем сумму отрицательных элементов на главной диагонали

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

if (matrix[i][i] < 0) {

sum += matrix[i][i];

}

}

cout << "Сумма отрицательных элементов на главной диагонали: " << sum << endl;

return 0;

}

\*\*\*\*\*

\*\*\*\*\*