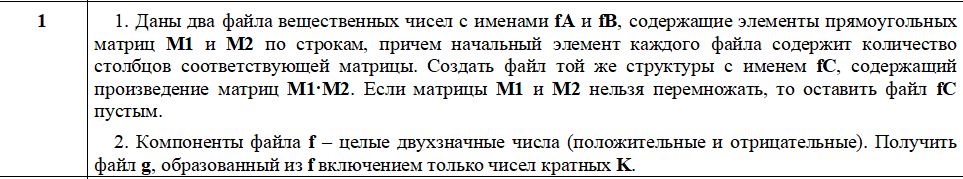
Банкузов Михаил группа 7 Лабораторная работа №2 Вариант 1



Задание 1

//Игнорируем ошибку т.к. fscanf не оптимизирована для новых компиляторов

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

int main() {

// Инициализация названий файлов

char filenameA[] = "fA.txt";

char filenameB[] = "fB.txt";

char filenameC[] = "fC.txt";

// Открытие файла A для чтения

FILE\* fileA = fopen(filenameA, "r");

if (fileA == NULL) {

printf("Не удалось открыть файл %s\n", filenameA);

return 1;

}

// Чтение количества столбцов в матрице A

int num\_colsA;

fscanf(fileA, "%d", &num\_colsA);

// Инициализация матрицы A

int num\_rowsA = 0;

double\* matrixA = NULL;

double value;

while (fscanf(fileA, "%lf", &value) == 1) {

num\_rowsA++;

matrixA = (double\*)realloc(matrixA, num\_rowsA \* num\_colsA \* sizeof(double));

matrixA[(num\_rowsA - 1) \* num\_colsA] = value;

for (int j = 1; j < num\_colsA; j++) {

if (fscanf(fileA, "%lf", &value) != 1) {

printf("Некорректный формат файла %s\n", filenameA);

free(matrixA);

return 1;

}

matrixA[(num\_rowsA - 1) \* num\_colsA + j] = value;

}

}

// Закрытие файла A

fclose(fileA);

// Открытие файла B для чтения

FILE\* fileB = fopen(filenameB, "r");

if (fileB == NULL) {

printf("Не удалось открыть файл %s\n", filenameB);

free(matrixA);

return 1;

}

// Чтение количества столбцов в матрице B

int num\_colsB;

fscanf(fileB, "%d", &num\_colsB);

// Инициализация матрицы B

int num\_rowsB = 0;

double\* matrixB = NULL;

while (fscanf(fileB, "%lf", &value) == 1) {

num\_rowsB++;

matrixB = (double\*)realloc(matrixB, num\_rowsB \* num\_colsB \* sizeof(double));

matrixB[(num\_rowsB - 1) \* num\_colsB] = value;

for (int j = 1; j < num\_colsB; j++) {

if (fscanf(fileB, "%lf", &value) != 1) {

printf("Некорректный формат файла %s\n", filenameB);

free(matrixA);

free(matrixB);

return 1;

}

matrixB[(num\_rowsB - 1) \* num\_colsB + j] = value;

}

}

// Закрытие файла B

fclose(fileB);

// Проверка возможности перемножения матриц

if (num\_colsA != num\_rowsB) {

printf("Невозможно перемножить матрицы\n");

free(matrixA);

free(matrixB);

return 0;

}

// Проверка возможности перемножения матриц

int num\_rowsC = num\_rowsA;

int num\_colsC = num\_colsB;

double\* matrixC = (double\*)calloc(num\_rowsC \* num\_colsC, sizeof(double));

for (int i = 0; i < num\_rowsC; i++) {

for (int j = 0; j < num\_colsC; j++) {

for (int k = 0; k < num\_colsA; k++) {

matrixC[i \* num\_colsC + j] += matrixA[i \* num\_colsA + k] \* matrixB[k \* num\_colsB + j];

}

}

}

// Открытие файла C для записи

FILE\* fileC = fopen(filenameC, "w");

if (fileC == NULL) {

printf("Не удалось открыть файл %s\n", filenameC);

return 1;

}

// Записываем результат в файл C

for (int i = 0; i < num\_rowsA; i++) {

for (int j = 0; j < num\_colsB; j++) {

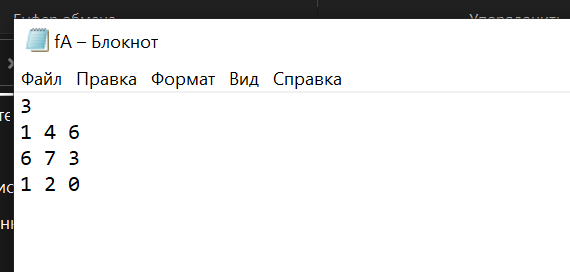
fprintf(fileC, "%.2lf ", matrixC[i \* num\_colsB + j]);

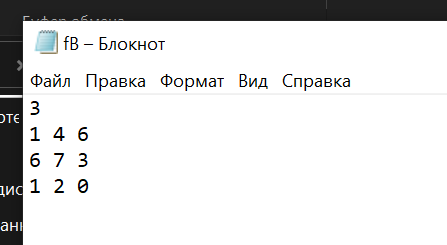
}

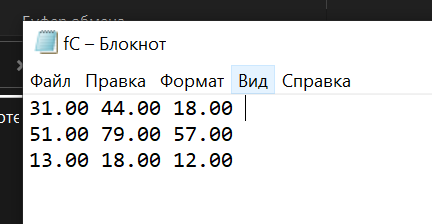
fprintf(fileC, "\n");

}

}







Задание 2

#include <iostream>

#include <fstream>

using namespace std;

int main() {

setlocale(LC\_ALL, "Russian");

// Открываем файл f для чтения

ifstream fileF("f.txt");

// Открываем файл g для записи

ofstream fileG("g.txt");

// Проверяем, удалось ли открыть файл f

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

cout << "Не удалось открыть файл f" << endl;

return 1;

}

// Проверяем, удалось ли открыть файл g

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

cout << "Не удалось открыть файл g" << endl;

return 1;

}

int k;

cout << "Введите число K: ";

cin >> k;

int number;

while (fileF >> number) {

// Если число не кратно K, пропускаем его

if (number % k != 0) {

continue;

}

// Иначе записываем его в файл g

fileG << number << " ";

}

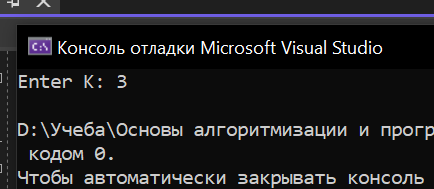
// Закрываем файлы

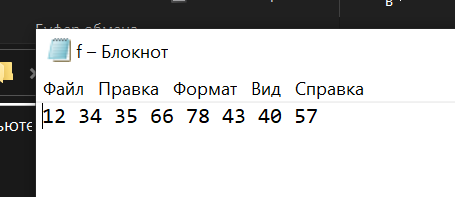
fileF.close();

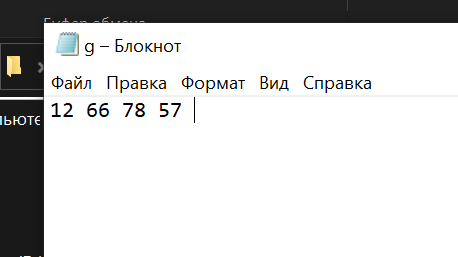
fileG.close();

return 0;

}

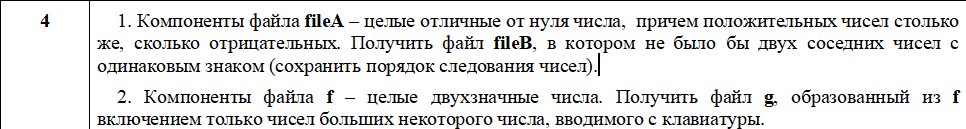






Дополнительные задания

Вариант 4



Задание 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

int main() {

// Открываем файлы fileA и fileB для чтения и записи соответственно

FILE\* fileA = fopen("fileA.txt", "r");

FILE\* fileB = fopen("fileB.txt", "w");

// Проверяем, удалось ли открыть файлы

if (!fileA) {

printf("Не удалось открыть файл fileA\n");

return 1;

}

if (!fileB) {

printf("Не удалось открыть файл fileB\n");

return 1;

}

int prevNumber = 0; // Переменная для хранения предыдущего числа

int currentNumber; // Переменная для хранения текущего числа

while (fscanf(fileA, "%d", &currentNumber) == 1) {

// Если предыдущее и текущее число имеют одинаковый знак, пропускаем текущее число

if ((prevNumber > 0 && currentNumber > 0) || (prevNumber < 0 && currentNumber < 0)) {

continue;

}

// Записываем текущее число в файл fileB

fprintf(fileB, "%d ", currentNumber);

// Сохраняем текущее число как предыдущее для следующей итерации

prevNumber = currentNumber;

}

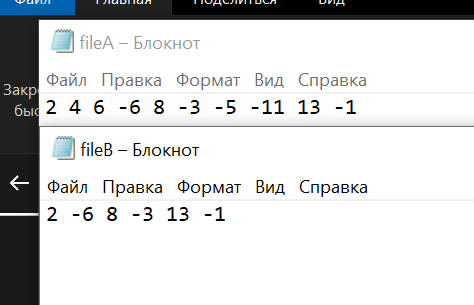
// Закрываем файлы

fclose(fileA);

fclose(fileB);

return 0;

}



Задание 2

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

int main() {

// Открываем файлы f и g

FILE\* fileF = fopen("f.txt", "r");

FILE\* fileG = fopen("g.txt", "w");

// Проверяем, удалось ли открыть файлы f и g

if (!fileF) {

printf("Не удалось открыть файл f\n");

return 1;

}

if (!fileG) {

printf("Не удалось открыть файл g\n");

return 1;

}

int num;

printf("Enter number: ");

scanf("%d", &num);

int number;

while (fscanf(fileF, "%d", &number) == 1) {

if (number > num && number > 9 && number < 100) {

fprintf(fileG, "%d\n", number);

}

}

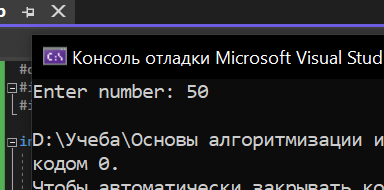
// Закрываем файлы f и g

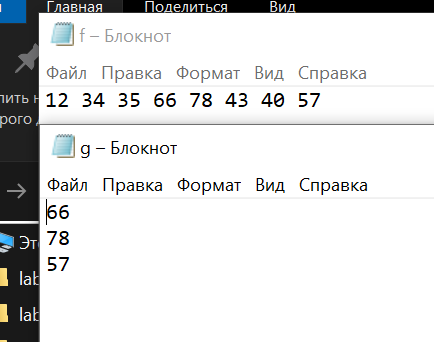
fclose(fileF);

fclose(fileG);

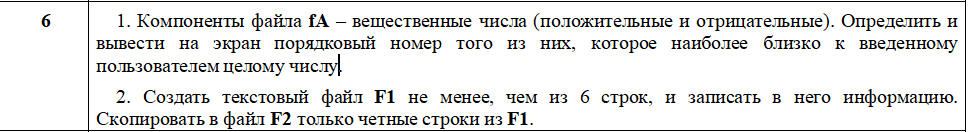
return 0;

}





Вариант 6



Задание 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <iostream>

int main() {

setlocale(LC\_ALL, "Russian");

// Открываем файл fA для чтения

FILE\* fileA = fopen("fA.txt", "r");

// Проверяем, удалось ли открыть файл fA

if (!fileA) {

printf("Не удалось открыть файл fA\n");

return 1;

}

// Считываем вещественные числа из файла fA

double numbers[100]; // предполагаем, что в файле не более 100 чисел

int numNumbers = 0;

double number;

while (fscanf(fileA, "%lf", &number) == 1) {

numbers[numNumbers] = number;

numNumbers++;

}

// Закрываем файл fA

fclose(fileA);

// Считываем целое число от пользователя

int target;

printf("Введите целое число: ");

scanf("%d", &target);

// Ищем число, наиболее близкое к целому числу, введенному пользователем

double closest = numbers[0];

int closestIndex = 0;

double diff = fabs(target - closest);

for (int i = 1; i < numNumbers; i++) {

double newDiff = fabs(target - numbers[i]);

if (newDiff < diff) {

closest = numbers[i];

closestIndex = i;

diff = newDiff;

}

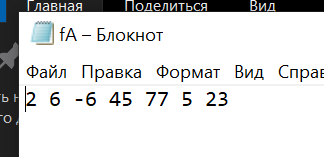
}

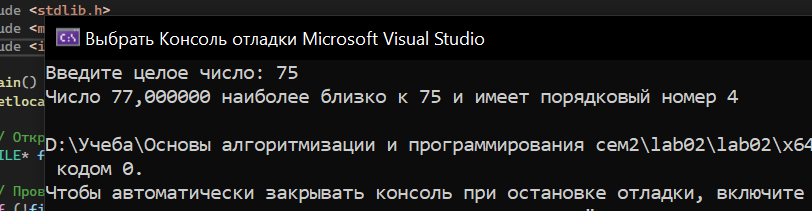
// Выводим порядковый номер найденного числа

printf("Число %lf наиболее близко к %d и имеет порядковый номер %d\n", closest, target, closestIndex);

return 0;

}





Задание 2

#define \_CRT\_SECURE\_NO\_WARNINGS

#define MAX\_LENGTH 1000

#include <stdio.h>

#include <stdlib.h>

#include <iostream>

int main() {

setlocale(LC\_ALL, "Russian");

// Открываем файл F1 для чтения

FILE\* file1 = fopen("f1.txt", "r");

// Открываем файл F2 для записи

FILE\* file2 = fopen("f2.txt", "w");

// Проверяем, удалось ли открыть файлы F1 и F2

if (!file1) {

printf("Не удалось открыть файл F1\n");

return 1;

}

if (!file2) {

printf("Не удалось открыть файл F2\n");

return 1;

}

// Считываем содержимое файла F1 построчно

char line[MAX\_LENGTH];

int lineNumber = 1;

while (fgets(line, MAX\_LENGTH, file1)) {

// Если номер строки четный, записываем ее в файл F2

if (lineNumber % 2 == 0) {

fprintf(file2, "%s", line);

}

lineNumber++;

}

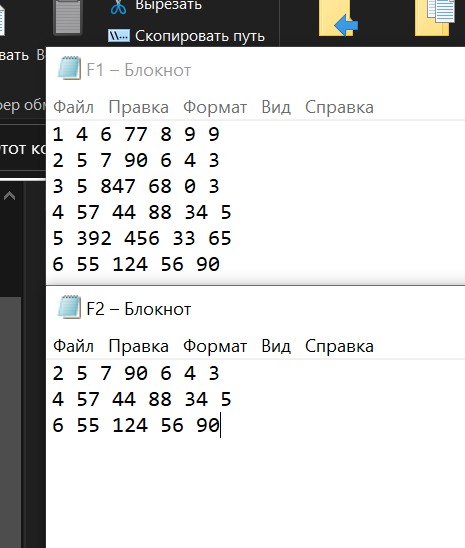
// Закрываем файлы

fclose(file1);

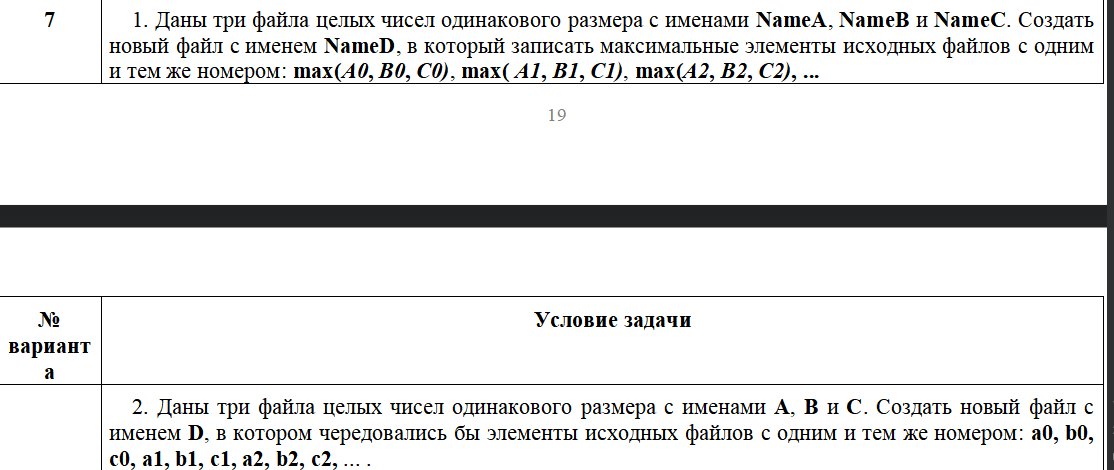
fclose(file2);

return 0;

}



Вариант 7



Задание 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <stdio.h>

#include <sstream>

#include <string>

#include <math.h>

#include <algorithm>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "Russian");

FILE\* NameA, \* NameB, \* NameC, \* NameD;

// Открываем три файла для чтения

NameA = fopen("NameA.txt", "r");

NameB = fopen("NameB.txt", "r");

NameC = fopen("NameC.txt", "r");

char arrA[10], arrB[10], arrC[10];

int arrD[4];

fgets(arrA, 10, NameA); //чтение строки данных

fgets(arrB, 10, NameB);

fgets(arrC, 10, NameC);

int temp;

int max\_el;

// Открываем файл для записи

NameD = fopen("NameD.txt", "w");

for (int i = 0; i < 4; i++)

{

// Переводим в числа и ищем максимальное

temp = max(stoi(arrA + i \* 2), stoi(arrB + i \* 2));

max\_el = max(temp, stoi(arrC + i \* 2));

cout << max\_el << " ";

arrD[i] = max\_el;

fprintf(NameD, "%d ", arrD[i]); //запись числа

}

// Закрываем файлы

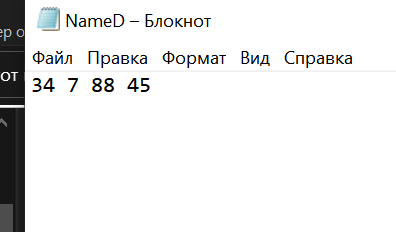
fclose(NameD);

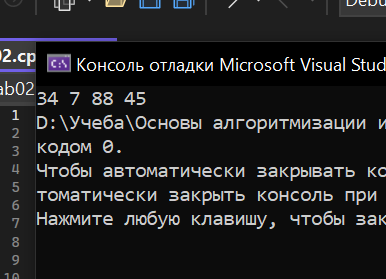
fclose(NameA);

fclose(NameB);

fclose(NameC);

}





Задание 2

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <stdio.h>

#include <sstream>

#include <string>

#include <math.h>

#include <algorithm>

using namespace std;

int main()

{

setlocale(LC\_CTYPE, "Russian");

FILE\* A, \* B, \* C, \* D;

// Открываем файлы для чтения

A = fopen("A.txt", "r");

B = fopen("B.txt", "r");

C = fopen("C.txt", "r");

char arrA[10], arrB[10], arrC[10];

int arrD[18];

fgets(arrA, 10, A); //чтение строки данных

fgets(arrB, 10, B);

fgets(arrC, 10, C);

int temp, element;

// Открываем файл для записи

D = fopen("D.txt", "w");

for (int i = 0; i < 4; i++)

{

temp = stoi(arrA + i \* 2); //перевод из символов в числа

element = temp;

cout << element << " ";

arrD[i] = element;

fprintf(D, "%d ", arrD[i]); //запись числа

for (int j = i; j < 4; j++)

{

temp = stoi(arrB + j \* 2);

element = temp;

cout << element << " ";

arrD[j] = element;

fprintf(D, "%d ", arrD[j]); //запись числа

for (int k = i; k < 4; k++)

{

temp = stoi(arrC + k \* 2);

element = temp;

cout << element << " ";

arrD[k] = element;

fprintf(D, "%d ", arrD[k]);//запись числа

break;

}

break;

}

}

// Закрываем файлы

fclose(D);

fclose(A);

fclose(B);

fclose(C);

}

