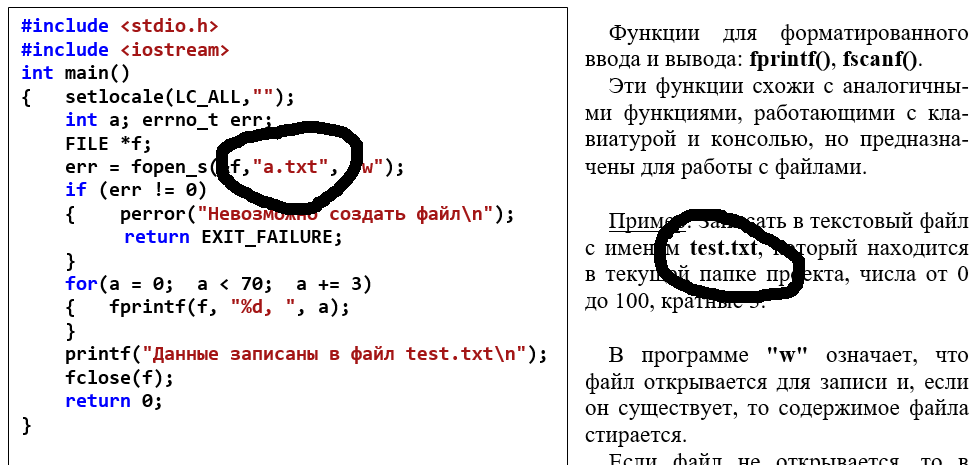
Задание 1



Мы работаем с файлом a.txt, а не с test.txt как написано в условии. Также при выполнение этого задания возникла ошибка C4996 ( она возникает в Visual Studio, когда мы используете функции, которые Microsoft считает небезопасными. В данном случае это касается функции fscanf. Поэтому я заменил её на fscanf\_s.

Изначальный вариант:

|  |  |  |
| --- | --- | --- |
| #include <stdio.h>  #include <iostream>  int main()  {  setlocale(LC\_ALL, "");  int a; errno\_t err;  FILE\* f;  err = fopen\_s(&f, "a.txt", "w");  if (err != 0)  {  perror("Невозможно создать файл\n");  return EXIT\_FAILURE;  }  for (a = 0; a < 70; a += 3)  {  fprintf(f, "%d, ", a);  }  printf("Данные записаны в файл a.txt\n");  fclose(f);  return 0;  }   |  |  | | --- | --- | |  |  | |

Дополненный вариант:

|  |  |  |
| --- | --- | --- |
| #include <stdio.h>  #include <iostream>  #include <locale.h>  int main()  {  setlocale(LC\_ALL, "");  int a;  errno\_t err;  FILE\* f;  err = fopen\_s(&f, "a.txt", "w");  if (err != 0)  {  perror("Невозможно создать файл\n");  return EXIT\_FAILURE;  }  for (a = 0; a < 70; a += 3)  {  fprintf(f, "%d, ", a);  }  fclose(f);  printf("Данные записаны в файл a.txt\n");  err = fopen\_s(&f, "a.txt", "r");  if (err != 0)  {  perror("Невозможно открыть файл для чтения\n");  return EXIT\_FAILURE;  }  printf("Данные из файла:\n");  while (fscanf\_s(f, "%d, ", &a) != EOF)  {  printf("%d\n", a);  }  fclose(f);  return 0;  }   |  |  | | --- | --- | |  |  | |

Задание 2

Изменил gets на безопасную версию fgets.

Изначальный вариант:

|  |  |  |
| --- | --- | --- |
| #include <iostream>  #include <stdio.h>  int main()  {  setlocale(LC\_ALL, "");  FILE\* mf; char str[50]; errno\_t err;  char\* estr;  printf("Введите текст "); gets\_s(str);  fopen\_s(&mf, "tst.txt", "w"); //Открытие для записи  fputs(str, mf); fputs("\n", mf);  fclose(mf);  err = fopen\_s(&mf, "tst.txt", "r"); //Открытие для чтения  if (err != NULL) //Проверка открытия файла  {  printf("Ошибка открытия файла\n"); return -1;  }  else  printf("\nСчитаны строки:\n");  estr = fgets(str, sizeof(str), mf); //Чтение строки файла  if (estr == NULL) //Конец файла или ошибка чтения?  if (feof(mf) != 0)  printf("\nЧтение файла закончено\n");  else  {  printf("\nОшибка чтения из файла\n");  return -1;  }  puts(str);  if (fclose(mf) == EOF)  printf("Ошибка закрытия\n");  return 0;  }   |  |  | | --- | --- | |  |  | |

Дополненный вариант:

|  |
| --- |
| он и так сделан выше |

Задание 3

Изначальный вариант:

|  |  |  |
| --- | --- | --- |
| #include <iostream>  #include <stdio.h>  int main(void)  {  setlocale(LC\_CTYPE, "Russian");  FILE\* fp;  errno\_t err;  char const\* st = "привет";  err = fopen\_s(&fp, "a.bin", "w");  if (err != 0)  {  perror("ошибка открытия a.txt");  return EXIT\_FAILURE;  }  fwrite(st, strlen(st), 1, fp);  printf("Записан элемент\n");  fclose(fp);  return 0;  }   |  |  | | --- | --- | |  |  | |

Дополненный вариант:

|  |  |  |
| --- | --- | --- |
| #include <iostream>  #include <stdio.h>  int main(void)  {  setlocale(LC\_CTYPE, "Russian");  FILE\* fp;  errno\_t err;  char const\* st = "привет";  err = fopen\_s(&fp, "a.bin", "w");  if (err != 0)  {  perror("ошибка открытия a.bin");  return EXIT\_FAILURE;  }  fwrite(st, strlen(st), 1, fp);  printf("Записан элемент\n");  fclose(fp);  err = fopen\_s(&fp, "a.bin", "r");  if (err != 0)  {  perror("Ошибка открытия a.bin");  return EXIT\_FAILURE;  }  char buffer[50];  size\_t count = fread(buffer, 1, sizeof(buffer) - 1, fp);//отнимаем 1 т.к. ещё будет записан \0, а так бы вышли за предел массива  buffer[count] = '\0';  printf("Прочитано из файла a.bin: %s\n", buffer);  fclose(fp);  return 0;  }   |  |  | | --- | --- | |  |  | |

Задание 4

|  |
| --- |
| err = fopen\_s(&fd, "b.txt", "r");  if (err != 0)  {  perror("ошибка открытия a.txt");  return;  } Изменил b.txt и a.txt на b.bin , чтобы соответствовало условию |

Изначальный вариант:

|  |  |  |
| --- | --- | --- |
| #include <iostream>  using namespace std;  void main()  {  setlocale(LC\_CTYPE, "Russian");  int m = 0, number = 0, pr = 0;  long fsize;  char pd;  FILE\* fd;  errno\_t err;  err = fopen\_s(&fd, "b.bin", "r");  if (err != 0)  {  perror("ошибка открытия b.bin");  return;  }  fseek(fd, 0L, SEEK\_END);// тут мы переместили указатель в конец  fsize = ftell(fd);//т.к. указатель в конце, то находим размер файла  fseek(fd, 0L, SEEK\_SET);//опять в начало переместили  printf("Введите номер предложения ");  scanf\_s("%d", &number);  for (int k = 1; k <= fsize; k++)  {  fread((void\*)&pd, sizeof(char), 1, fd);  if (pd == '.') pr++;  if ((number - 1) == pr) m++;  if (number == pr)  {  long pos1 = k - m - 1;  if (number != 1) pos1++;  fseek(fd, pos1, SEEK\_SET);  printf("%d-е предложение: ", number);  for (int i = 0; i <= m; i++)  {  fread((void\*)&pd, sizeof(char), 1, fd);  printf("%c", pd);  }  break;  }  }  if (number > pr)  printf("Такого номера нет");  fclose(fd);  }   |  |  | | --- | --- | |  |  | |

Дополненный вариант:

|  |  |  |
| --- | --- | --- |
| #include <iostream>  using namespace std;  void main()  {  setlocale(LC\_CTYPE, "Russian");  int m = 0, pr = 0;  long fsize;  char pd;  FILE\* fd;  errno\_t err;  err = fopen\_s(&fd, "b.bin", "r");  if (err != 0)  {  perror("ошибка открытия b.bin");  return;  }  fseek(fd, 0L, SEEK\_END);// тут мы переместили указатель в конец  fsize = ftell(fd);//т.к. указатель в конце, то находим размер файла  fseek(fd, 0L, SEEK\_SET);//опять в начало переместили  int number;  printf("Введите количество предложений для вывода: ");  scanf\_s("%d", &number);  printf("Предложения с 1 по %d:\n", number);  for (int k = 1; k <= fsize; k++)  {  fread((void\*)&pd, sizeof(char), 1, fd);  if (pd == '.')  {  pr++;  m++;  if (pr <= number)  printf("%c", pd);  if (pr == number)  break;  }  else if (pr < number)  {  m++;  printf("%c", pd);  }  }  if (number > pr)  printf("Такого номера нет");  fclose(fd);  }   |  |  | | --- | --- | |  |  | |

Вариант 12

|  |
| --- |
| 1. Дан файл вещественных чисел, содержащий элементы квадратной матрицы по строкам, причем начальный элемент файла содержит значение количества столбцов матрицы. Создать новый файл той же структуры, содержащий **k**-ую строку исходной матрицы. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<Windows.h>  void row(const char\* inputFile, const char\* outputFile, int k) {  FILE\* input;  FILE\* output;  errno\_t errInput, errOutput;  errInput = fopen\_s(&input, inputFile, "r");  if (errInput != 0 ) {  perror("Ошибка открытия входного файла");  exit(EXIT\_FAILURE);  }  errOutput = fopen\_s(&output, outputFile, "w");  if (errOutput != 0 ) {  perror("Ошибка открытия выходного файла");  fclose(input);  exit(EXIT\_FAILURE);  }  int cols;  if (fscanf\_s(input, "%d", &cols) != 1) {  printf("Ошибка чтения количества столбцов\n");  fclose(input);  fclose(output);  exit(EXIT\_FAILURE);  }  if (cols <= 0) {  printf("Некорректное количество столбцов\n");  fclose(input);  fclose(output);  exit(EXIT\_FAILURE);  }  if (k < 1) {  printf("Некорректный номер строки\n");  fclose(input);  fclose(output);  exit(EXIT\_FAILURE);  }  double value;  int currentRow = 1;  int found = 0;  while (!feof(input)) {  for (int col = 1; col <= cols; ++col) {  if (fscanf\_s(input, "%lf", &value) == 1) {  if (currentRow == k) {  fprintf(output, "%.2lf ", value);  found = 1;  }  } else {  break;  }  }  if (currentRow == k) {  break;  }  currentRow++;  }  if (!found) {  printf("Строка %d не найдена\n", k);  } else {  printf("Строка %d успешно записана в файл %s\n", k, outputFile);  }  fclose(input);  fclose(output);  }  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  const char\* inputFile = "matrix.txt";  const char\* outputFile = "output.txt";  int k;  printf("Введите номер строки для извлечения: ");  scanf\_s("%d", &k);  row(inputFile, outputFile, k);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |  |  | |

|  |
| --- |
| 2. Даны три файла целых чисел одинакового размера с именами **NameA**, **NameB** и **NameC**. Создать новый файл с именем **NameD**, в который записать минимальные элементы исходных файлов с одним и тем же номером: **min(*A0*, *B0*, *C0)***, **min( *A1*, *B1*, *C1)***, **min(*A2*, *B2*, *C2)*, ...** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<iostream>  void getMin(const char\* NameA, const char\* NameB, const char\* NameC, const char\* NameD) {  FILE\* fileA, \* fileB, \* fileC, \* fileD;  errno\_t errA, errB, errC, errD;  errA = fopen\_s(&fileA, NameA, "r");  errB = fopen\_s(&fileB, NameB, "r");  errC = fopen\_s(&fileC, NameC, "r");  errD = fopen\_s(&fileD, NameD, "w");  if (errA != 0 || errB != 0 || errC != 0 || errD != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int a, b, c;  while (fscanf\_s(fileA, "%d", &a) == 1 && fscanf\_s(fileB, "%d", &b) == 1 && fscanf\_s(fileC, "%d", &c) == 1) {  int minimum;  if (a < b) {  if (a < c) {  minimum = a;  }  else {  minimum = c;  }  }  else {  if (b < c) {  minimum = b;  }  else {  minimum = c;  }  }  fprintf(fileD, "%d\n", minimum);  }  fclose(fileA);  fclose(fileB);  fclose(fileC);  fclose(fileD);  printf("Файл %s успешно создан.\n", NameD);  }  int main() {  setlocale(LC\_ALL, "ru");  const char\* NameA = "NameA.txt";  const char\* NameB = "NameB.txt";  const char\* NameC = "NameC.txt";  const char\* NameD = "NameD.txt";  getMin(NameA, NameB, NameC, NameD);  FILE\* fileD;  errno\_t errD = fopen\_s(&fileD, NameD, "r");  if (errD != 0) {  perror("Ошибка открытия файла NameD");  return EXIT\_FAILURE;  }  printf("Содержимое файла %s:\n", NameD);  int x;  while (fscanf\_s(fileD, "%d", &x) == 1) {  printf("%d ", x);  }  printf("\n");  fclose(fileD);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |  |  | |

Доп.варианты:

Вариант 1 (12+5=17=1)

|  |
| --- |
| 1. Даны два файла вещественных чисел с именами **fA** и **fB**, содержащие элементы прямоугольных матриц **M1** и **M2** по строкам, причем начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем **fC**, содержащий произведение матриц **M1·M2**. Если матрицы **M1** и **M2** нельзя перемножать, то оставить файл **fC** пустым. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<Windows.h>  void matrices(const char\* fA, const char\* fB, const char\* fC) {  FILE\* fileA, \* fileB, \* fileC;  errno\_t errA, errB, errC;  errA = fopen\_s(&fileA, fA, "r");  errB = fopen\_s(&fileB, fB, "r");  errC = fopen\_s(&fileC, fC, "w");  if (errA != 0 || errB != 0 || errC != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int n1, m1, n2, m2;  if (fscanf\_s(fileA, "%d", &n1) != 1) {  printf("Ошибка чтения количества столбцов для матрицы M1\n");  fclose(fileA);  fclose(fileB);  fclose(fileC);  return;  }  if (fscanf\_s(fileA, "%d", &m1) != 1) {  printf("Ошибка чтения количества строк для матрицы M1\n");  fclose(fileA);  fclose(fileB);  fclose(fileC);  return;  }  if (fscanf\_s(fileB, "%d", &n2) != 1) {  printf("Ошибка чтения количества столбцов для матрицы M2\n");  fclose(fileA);  fclose(fileB);  fclose(fileC);  return;  }  if (fscanf\_s(fileB, "%d", &m2) != 1) {  printf("Ошибка чтения количества строк для матрицы M2\n");  fclose(fileA);  fclose(fileB);  fclose(fileC);  return;  }  if (n1 != m2) {  printf("Матрицы нельзя перемножить. Количество столбцов первой матрицы не равно количеству строк второй матрицы.\n");  fclose(fileA);  fclose(fileB);  fclose(fileC);  return;  }  double\*\* M1 = (double\*\*)malloc(m1 \* sizeof(double\*));  for (int i = 0; i < m1; i++) {  M1[i] = (double\*)malloc(n1 \* sizeof(double));  for (int j = 0; j < n1; j++) {  fscanf\_s(fileA, "%lf", &M1[i][j]);  }  }  double\*\* M2 = (double\*\*)malloc(m2 \* sizeof(double\*));  for (int i = 0; i < m2; i++) {  M2[i] = (double\*)malloc(n2 \* sizeof(double));  for (int j = 0; j < n2; j++) {  fscanf\_s(fileB, "%lf", &M2[i][j]);  }  }  double\*\* M3 = (double\*\*)malloc(m1 \* sizeof(double\*));  for (int i = 0; i < m1; i++) {  M3[i] = (double\*)malloc(n2 \* sizeof(double));  for (int j = 0; j < n2; j++) {  M3[i][j] = 0;  for (int k = 0; k < n1; k++) {  M3[i][j] += M1[i][k] \* M2[k][j];  }  }  }  fprintf(fileC, "%d\n", n2);  fprintf(fileC, "%d\n", m1);  for (int i = 0; i < m1; i++) {  for (int j = 0; j < n2; j++) {  fprintf(fileC, "%.2lf ", M3[i][j]);  }  fprintf(fileC, "\n");  }  for (int i = 0; i < m1; i++) {  free(M1[i]);  }  free(M1);  for (int i = 0; i < m2; i++) {  free(M2[i]);  }  free(M2);  for (int i = 0; i < m1; i++) {  free(M3[i]);  }  free(M3);  fclose(fileA);  fclose(fileB);  fclose(fileC);  printf("Файл %s успешно создан.\n", fC);  }  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  const char\* fA = "fA.txt";  const char\* fB = "fB.txt";  const char\* fC = "fC.txt";  matrices(fA, fB, fC);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |

|  |
| --- |
| 2. Компоненты файла **f** – целые двухзначные числа (положительные и отрицательные). Получить файл **g**, образованный из **f** включением только чисел кратных **K**. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<Windows.h>  void fK(const char\* inputFile, const char\* outputFile, int K) {  FILE\* f, \* g;  errno\_t errF, errG;  errF = fopen\_s(&f, inputFile, "r");  errG = fopen\_s(&g, outputFile, "w");  if (errF != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int number;  while (fscanf\_s(f, "%d", &number) == 1) {  if (number % K == 0) {  fprintf(g, "%d\n", number);  }  }  fclose(f);  fclose(g);  printf("Файл %s успешно создан с числами, кратными %d.\n", outputFile, K);  }  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  const char\* inputFile = "f.txt";  const char\* outputFile = "g.txt";  int K;  printf("Введите число K: ");  scanf\_s("%d", &K);  if (K == 0) {  printf("Ошибка: K не может быть равно 0.\n");  return EXIT\_FAILURE;  }  fK(inputFile, outputFile, K);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |

Вариант 7(12-5=7)

|  |
| --- |
| 1. Даны три файла целых чисел одинакового размера с именами **NameA**, **NameB** и **NameC**. Создать новый файл с именем **NameD**, в который записать максимальные элементы исходных файлов с одним и тем же номером: **max(*A0*, *B0*, *C0)***, **max( *A1*, *B1*, *C1)***, **max(*A2*, *B2*, *C2)*, ...** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<iostream>  void getMin(const char\* NameA, const char\* NameB, const char\* NameC, const char\* NameD) {  FILE\* fileA, \* fileB, \* fileC, \* fileD;  errno\_t errA, errB, errC, errD;  errA = fopen\_s(&fileA, NameA, "r");  errB = fopen\_s(&fileB, NameB, "r");  errC = fopen\_s(&fileC, NameC, "r");  errD = fopen\_s(&fileD, NameD, "w");  if (errA != 0 || errB != 0 || errC != 0 || errD != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int a, b, c;  while (fscanf\_s(fileA, "%d", &a) == 1 && fscanf\_s(fileB, "%d", &b) == 1 && fscanf\_s(fileC, "%d", &c) == 1) {  int max;  if (a > b) {  if (a > c) {  max = a;  }  else {  max = c;  }  }  else {  if (b > c) {  max = b;  }  else {  max = c;  }  }  fprintf(fileD, "%d\n", max);  }  fclose(fileA);  fclose(fileB);  fclose(fileC);  fclose(fileD);  printf("Файл %s успешно создан.\n", NameD);  }  int main() {  setlocale(LC\_ALL, "ru");  const char\* NameA = "NameA.txt";  const char\* NameB = "NameB.txt";  const char\* NameC = "NameC.txt";  const char\* NameD = "NameD.txt";  getMin(NameA, NameB, NameC, NameD);  FILE\* fileD;  errno\_t errD = fopen\_s(&fileD, NameD, "r");  if (errD != 0) {  perror("Ошибка открытия файла NameD");  return EXIT\_FAILURE;  }  printf("Содержимое файла %s:\n", NameD);  int x;  while (fscanf\_s(fileD, "%d", &x) == 1) {  printf("%d ", x);  }  printf("\n");  fclose(fileD);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |  |

|  |
| --- |
| 2. Даны три файла целых чисел одинакового размера с именами **A**, **B** и **C**. Создать новый файл с именем **D**, в котором чередовались бы элементы исходных файлов с одним и тем же номером: **a0, b0, c0, a1, b1, c1, a2, b2, c2,** ... . |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include <locale.h>  void mergeFiles(const char\* A, const char\* B, const char\* C, const char\* D) {  FILE\* fileA, \* fileB, \* fileC, \* fileD;  errno\_t errA, errB, errC, errD;  errA = fopen\_s(&fileA, A, "r");  errB = fopen\_s(&fileB, B, "r");  errC = fopen\_s(&fileC, C, "r");  errD = fopen\_s(&fileD, D, "w");  if (errA != 0 || errB != 0 || errC != 0 || errD != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int a, b, c;  while (fscanf\_s(fileA, "%d", &a) == 1 && fscanf\_s(fileB, "%d", &b) == 1 && fscanf\_s(fileC, "%d", &c) == 1) {  fprintf(fileD, "%d %d %d\n", a, b, c);  }  fclose(fileA);  fclose(fileB);  fclose(fileC);  fclose(fileD);  printf("Файл %s успешно создан.\n", D);  }  int main() {  setlocale(LC\_ALL, "ru");  const char\* NameA = "A.txt";  const char\* NameB = "B.txt";  const char\* NameC = "C.txt";  const char\* NameD = "D.txt";  mergeFiles(NameA, NameB, NameC, NameD);  FILE\* fileD;  errno\_t errD = fopen\_s(&fileD, NameD, "r");  if (errD != 0) {  perror("Ошибка открытия файла D");  return EXIT\_FAILURE;  }  printf("Содержимое файла %s:\n", NameD);  int x;  while (fscanf\_s(fileD, "%d", &x) == 1) {  printf("%d ", x);  }  printf("\n");  fclose(fileD);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |

Вариант 2(12+6=18=2)

|  |
| --- |
| 1. Даны два файла вещественных чисел с именами **fA** и **fB**, содержащие элементы прямоугольных матриц **M1** и **M2** по строкам, причем начальный элемент каждого файла содержит количество столбцов соответствующей матрицы. Создать файл той же структуры с именем **fC**, содержащий сумму **M1+M2**. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<Windows.h>  void matrices(const char\* fileA, const char\* fileB, const char\* fileC) {  FILE\* fA, \* fB, \* fC;  errno\_t errA, errB, errC;  errA = fopen\_s(&fA, fileA, "r");  errB = fopen\_s(&fB, fileB, "r");  errC = fopen\_s(&fC, fileC, "w");  if (errA != 0 || errB != 0 || errC != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int colsA, colsB, colsRemaining;  if (fscanf\_s(fA, "%d", &colsA) != 1 || fscanf\_s(fB, "%d", &colsB) != 1) {  printf("Ошибка чтения количества столбцов из файлов\n");  fclose(fA);  fclose(fB);  fclose(fC);  exit(EXIT\_FAILURE);  }  if (colsA != colsB) {  printf("Матрицы имеют разное количество столбцов, сложение невозможно\n");  fclose(fA);  fclose(fB);  fclose(fC);  exit(EXIT\_FAILURE);  }  fprintf(fC, "%d\n", colsA);  double valueA, valueB;  colsRemaining = colsA;  while (fscanf\_s(fA, "%lf", &valueA) == 1 && fscanf\_s(fB, "%lf", &valueB) == 1) {  fprintf(fC, "%.2lf ", valueA + valueB);  colsRemaining--;  if (colsRemaining == 0) {  fprintf(fC, "\n");  colsRemaining = colsA;  }  }  fclose(fA);  fclose(fB);  fclose(fC);  printf("Файл %s успешно создан.\n", fileC);  }  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  const char\* fileA = "fA.txt";  const char\* fileB = "fB.txt";  const char\* fileC = "fC.txt";  matrices(fileA, fileB, fileC);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |

|  |
| --- |
| 2. Компоненты файла**f** – целые двухзначные (отличные от нуля) числа, причем 10 положительных чисел, 10 отрицательных, и т. д. Получить файл **g***,* в котором записаны сначала 5 положительных чисел, затем 5 отрицательных и т. д. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| #include <stdio.h>  #include <stdlib.h>  #include<Windows.h>  #define SIZE 5  void pF(const char\* inputFile, const char\* outputFile) {  FILE\* fInput, \* fOutput;  errno\_t errInput, errOutput;  errInput = fopen\_s(&fInput, inputFile, "r");  errOutput = fopen\_s(&fOutput, outputFile, "w");  if (errInput != 0 || errOutput != 0) {  perror("Ошибка открытия файлов");  exit(EXIT\_FAILURE);  }  int numbers[1000];  int posBuffer[1000], negBuffer[1000];  int totalNumbers = 0, posCount = 0, negCount = 0;  while (fscanf\_s(fInput, "%d", &numbers[totalNumbers]) == 1) {  if (numbers[totalNumbers] > 0) {  posBuffer[posCount++] = numbers[totalNumbers];  }  else {  negBuffer[negCount++] = numbers[totalNumbers];  }  totalNumbers++;  }  int i = 0, j = 0;  while (i < posCount || j < negCount) {  for (int k = 0; k < SIZE && i < posCount; k++) {  fprintf(fOutput, "%d ", posBuffer[i++]);  }  for (int k = 0; k <SIZE && j < negCount; k++) {  fprintf(fOutput, "%d ", negBuffer[j++]);  }  }  fclose(fInput);  fclose(fOutput);  printf("Файл %s успешно создан.\n", outputFile);  }  int main() {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  const char\* inputFile = "f.txt";  const char\* outputFile = "g.txt";  pF(inputFile, outputFile);  return 0;  }   |  |  | | --- | --- | |  |  | |  |  | |