

«Разработка алгоритмов и программ с использованием библиотечных модулей пользователя»

Для учащегося предназначено 5 заданий, соответствующих его номеру по списку группы (по журналу класса).

После написания кода программы в функции `main()`, разбить его на смысловые блоки и на их основе создать функции пользователя. Далее все функции пользователя и их прототипы переместить в заголовочный файл `header.h`. Функции пользователя вызывать в функции `main()` с фактическими параметрами. Функция `main()` остается в `.cpp`-файле, в котором добавляется строка кода по подключению одного или нескольких заголовочных файлов `headerFamilia.h` в имя которого поместите вашу фамилию, например, `HeaderSidorov.h`.

Сделать 1 программу (1 проект), решающую все 5 задач вашего варианта. Проект должен состоять из заголовочного `headerВашаФамилия.h` файла и `.cpp`-файла.

Заголовочный файл – это файл с расширением `.h`, который применяется для хранения функций и их прототипов, переменных и констант. Заголовочный файл можно подключить к исходному файлу программы (`Source.cpp`) и вызывать из `main`-а любые функции, переменные и константы, которые есть в подключенном к нему заголовочном файле. Как результат, код исходного файла программы (`Source.cpp`) становится меньше, поскольку все функции, их прототипы, глобальные константы и переменные находятся в заголовочном файле. При этом исходный файл использует функции из заголовочного файла, а потому **зависит** от последнего (например, если удалить или переместить в другое место на ПК заголовочный файл, то исходный файл не сможет корректно проработать, ведь он вызывает функции из заголовочного файла, а его не окажется по прописанному пути).

Заголовочный файл назван от англ. «Header.h», что буквально переводится как «заголовочный, головной» (от англ. head – голова), то есть файл с заголовками (прототипами) функций, констант, переменных. Расширение «.h» происходит именно от сокращения английского слова «header».

Заголовочный файл можно рассматривать как простую библиотеку. Библиотека – это файл для хранения функций и их прототипов, классов и их членов, методов, переменных, констант и т.д. Библиотеки делятся на:

- 1) статические (имеют расширение «.lib» от англ. «library» – библиотека), которые подключаются ДО компиляции программы, а потому все их функции и прототипы будут заранее видны при написании программы, которая использует функции из подключенной к ней статической библиотеки и
- 2) динамические (имеют расширение «.dll» от англ. “dynamic link library” – динамически подключаемая библиотека), которые подключаются на этапе выполнения программы, то есть в момент вызова функций из них, в реальном времени, «на лету». DLL экономят ресурсы компьютера, вызываются только тогда, когда в них есть потребность, и сразу отсоединяются, когда вызванная из них функция отработала и больше не нужна. Но при написании программ, использующих dll, надо внимательно писать функции (их имена) и самому контролировать количество, порядок следования и типы передаваемых в функции из dll входных фактических параметров, ведь ошибки проявятся только при работе готовой программы, а не на этапе набора кода (компилятор не поможет).

Заголовочные файлы являются простейшими библиотеками, поскольку при подключении заголовочного файла **весь** его код помещается в вызывающий исходный файл, даже если в `main`-е из заголовочного файла будет вызвана только одна функция или не одной вообще. А из статических библиотек компоновщик помещает в программу только те функции, которые вызываются в `main`-е. Заголовочный файл не компилируется в отдельный бинарный файл как «настоящие» библиотеки. Но все же код заголовочного файла помещается в код вызвавшего его исходного файла, становится его частью и будет частью скомпилированного исполнимого бинарного `.exe`-файла.

Если в языке высокого уровня Си настоятельно рекомендуется писать прототипы для функций, поскольку написание прототипов упрощает контроль типов данных (типов переменных, передаваемых в функции), то в языке C++ написание прототипов обязательно по правилам языка и показывает знание разработчиком этой нормы языка. Прототипы пишутся вверху файла, а полное определение функций (сами функции с заголовком и телом) – внизу файла.

Чтобы использовать в `.cpp`-файле функции из заголовочного файла с именем `Header.h`, надо в начале `.cpp`-файла написать код подключения файла:

```
#include "Header.h"
```

Обратите внимание, что имя заголовочного файла, написанного разработчиком, берется в двойные кавычки, в то время как имена стандартных подключаемых библиотек записываются в угловых скобках:

```
#include <time.h>
```

Можно писать и полное имя заголовочного файла – это полный путь к файлу, начиная с указания буквы диска на ПК, далее папок и подпапок вплоть до собственно заголовочного файла. Это можно делать по отношению к любому заголовочному файлу, но если заголовочный файл находится вне папки проекта, где находится вызывающий его `.cpp`-файл, то необходимо прописать полный путь к заголовочному файлу:

```
#include "D:\\Folder1\\Subfolder2\\Subfolder3\\HeaderErmakov.h"
```

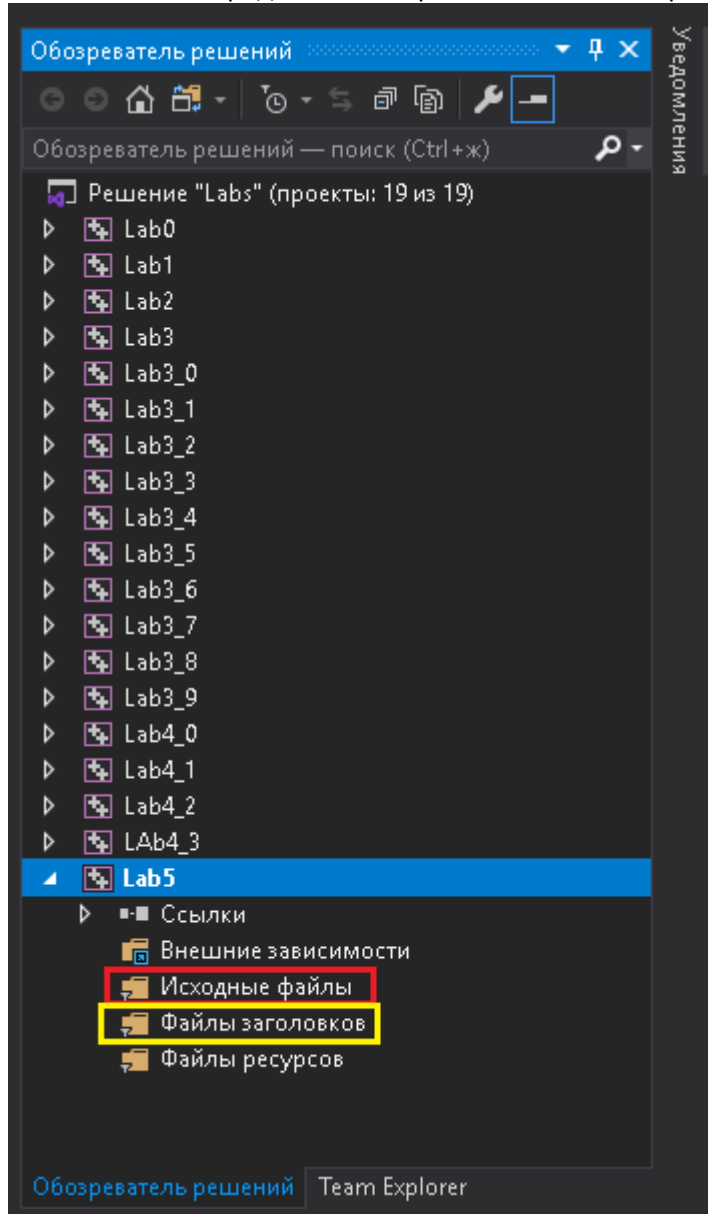
Есть возможность указать и относительный путь. Для этого нужно аккуратно последовательно набрать `#include “..\\` и далее ориентироваться на всплывающие подсказки от VS2019 с именами ближайших папок на данном ПК. Лучше всего располагать заголовочный файл в папке с проектом, где он используется. Всегда проверяйте, скопировали ли заголовочные файлы вместе с проектами (`.cpp`-файлами) на флешку для демонстрации в колледже,

поскольку, если в .cpp-файле используется хотя бы одна функция из заголовочного файла, а последнего нет (или путь к нему поменялся), то ваша программа **НЕ работоспособна**. Разбиение кода программы на несколько файлов может помочь структурировать программу, но это же порождает зависимости одних файлов от других.

Создадим проект с заголовочным файлом.

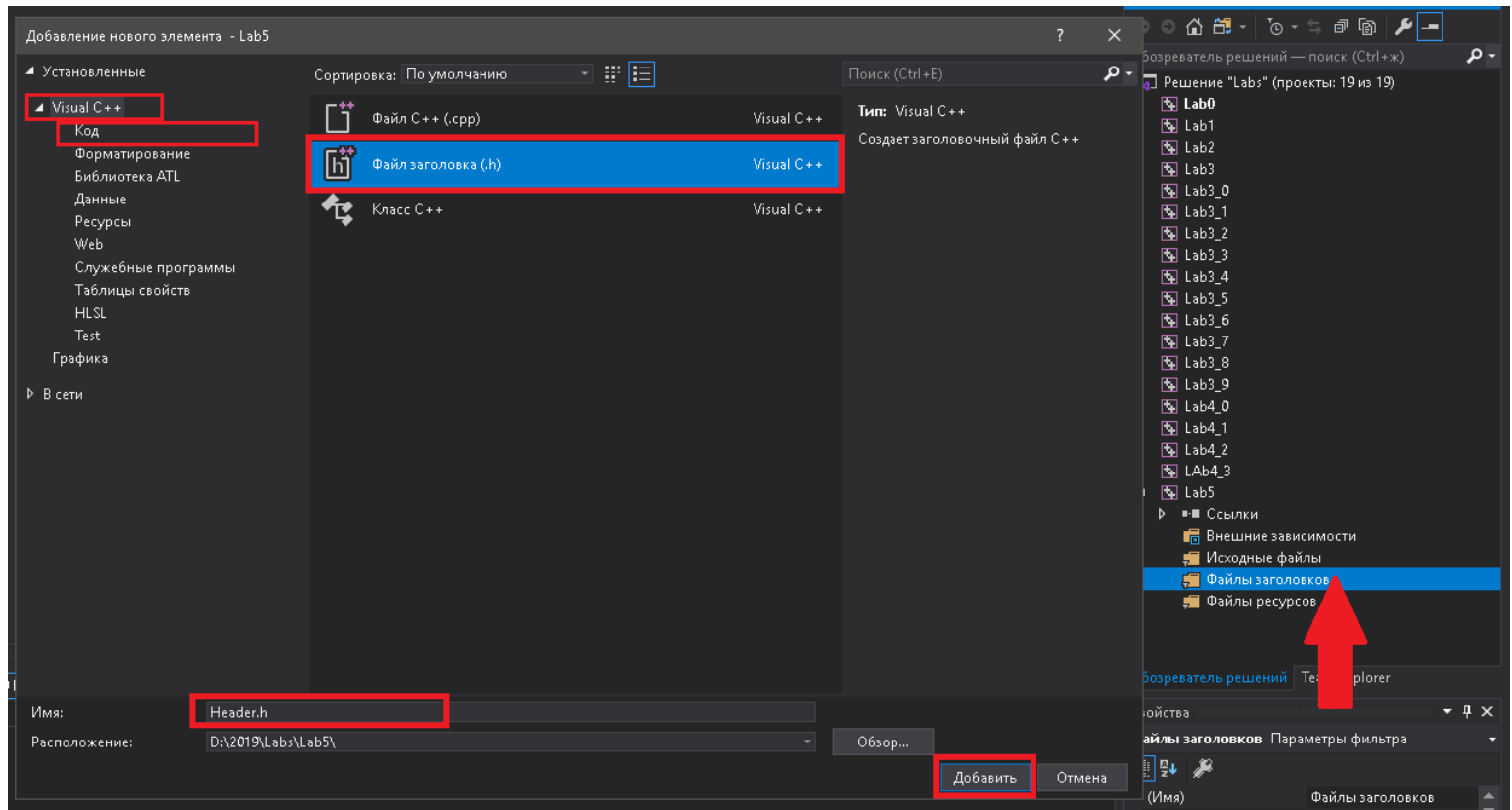
Изначальные действия такие же, как и обычные действия по созданию пустого C++ проекта. Обратите внимание, в окне Обозревателя решений в папке с проектом есть 3 подпапки:

- 1) Исходные файлы – сюда помещается код .cpp-файлов.
- 2) Файлы заголовков – специальная папка для хранения заголовочных файлов. Заголовочные файлы должны создаваться и храниться в ней.
- 3) Файлы ресурсов – пока не используемая нами папка для хранения используемых в проекте ресурсов, например, файлов-изображений, музыкальных-файлов, файлов с данными, файлов-«иконок» для приложения (это маленькая картинка, пиктограмма, значок для вашего приложения; если он не создан, то операционная система предоставляет приложениям пиктограмму по умолчанию).



Нужно сказать, что на самом деле в папке с проектом есть 3 файла с настройками нашего проекта на C++, но физически вложенных папок пока нет. Просто Обозреватель решений имеет фильтрацию, которая обеспечивает группировку файлов определенных расширений по группам. Эти 3 группы фильтрации Обозреватель решений отображает в виде «папок».

Кликнем правой кнопкой мыши по имени проекта или папке «Файлы заголовков», выберем в контекстном меню Добавить \ Создать элемент \ Visual C++ \ Код \ Файл заголовка (.h) \ Добавить. В поле «Имя» можно поменять стандартное имя Header.h на свое, например, HeaderPetrova.h.

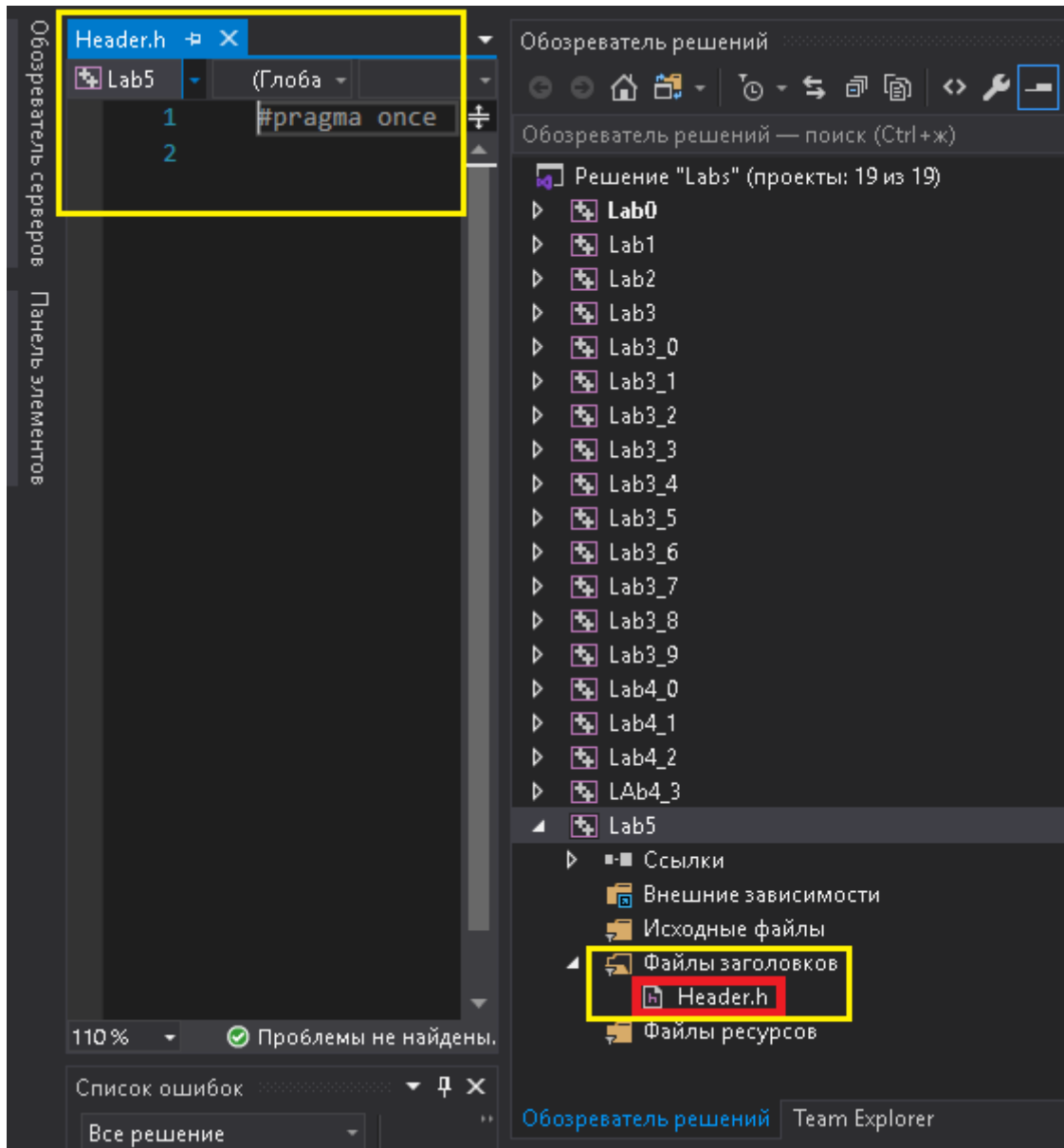


Заголовочный файл Header.h должен появиться в подпапке заголовочных файлов вашего проекта. В основном окне должен отобразиться этот заголовочный файл со своим содержимым – одной строкой кода:

#pragma once

что является директивой препроцессору, указывающей, что при повторении кода в этом файле и других (например, строки подключения библиотек, определения функций и их описания (прототипы)), следует считать данные функции, прототипы и т. д. как определенные единожды (определенные в программе только один раз).

Итак, заполняем заголовочный файл. Сначала подключаем в нем библиотеки, другие заголовочные файлы, пространства имен, которые требуются для работы функций данного заголовочного файла. Далее должны быть написаны прототипы функций, а за ними полные определения самих этих функций. В заголовочном файле не может находиться функция `main()` – она может и должна находиться в `.cpp`-файле данного проекта, причем только одна. Правило: один проект – один `main()`.



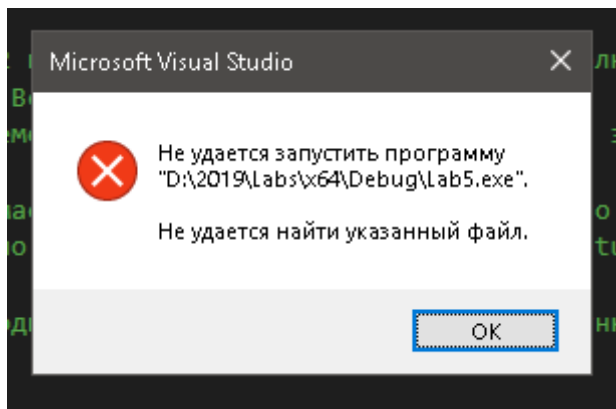
Итак, допишем заголовочный файл:

```

1  #pragma once
2  #include <iostream>
3  #include <cmath>
4  using namespace std;
5
6  void fillArrayFibonacci(int*, int); //прототип функции заполнения массива числами Фибоначчи
7  void printArray(int*, int); //прототип функции печати значений элементов одномерного целочисленного массива на консоль
8
9  void fillArrayFibonacci(int* m, int n) //функция заполнения целочисленного одномерного массива числами Фибоначчи
10 {
11     if (n < 2) //если в массиве меньше 2-х элементов
12     {
13         return; //делать нечего - досрочно завершаем функцию
14     }
15     m[0] = 0; //если выполняется эта строка кода, значит в массиве 2 или больше элементов - их можно заполнять числами Фибоначчи
16     m[1] = 1; //первые 2 элемента массива - это 0 и 1 по умолчанию. Все остальные можно вычислить
17     for (int i = 2; i < n; i++) //проходим по массиву, начиная с элемента с индексом № 2 и далее по всем, заканчивая индексом n-1 включительно
18     {
19         m[i] = m[i - 1] + m[i - 2]; //значение очередного элемента массива есть сумма значений предыдущего элемента и предшествовавшего предыдущему элементу массива
20     } //функция возвращает void - то есть ничего, пустоту. Также можно завершить эту функцию выражением return; //то есть "возвратить ничего"
21 }
22 void printArray(int* m, int n) //функция печати значений элементов одномерного целочисленного массива. Функция ничего не возвращает (void)
23 {
24     cout << "Array:\n"; //напечатаем на консоль предваряющую фразу
25     for (int i = 0; i < n; i++) //цикл для прохода по массиву от первого элемента с индексом № 0 до последнего элемента с индексом n-1
26     {
27         cout << m[i] << ' '; //печатаем значение элемента массива с индексом № i и один пробел за ним (пробел - 1 символ, можно взять в одинарные кавычки)
28     }
29     cout << endl; //переведем курсор на новую строку //функция возвращает void - то есть ничего, пустоту. Можно завершить эту функцию выражением return; //то есть "возвратить ничего"
30 }

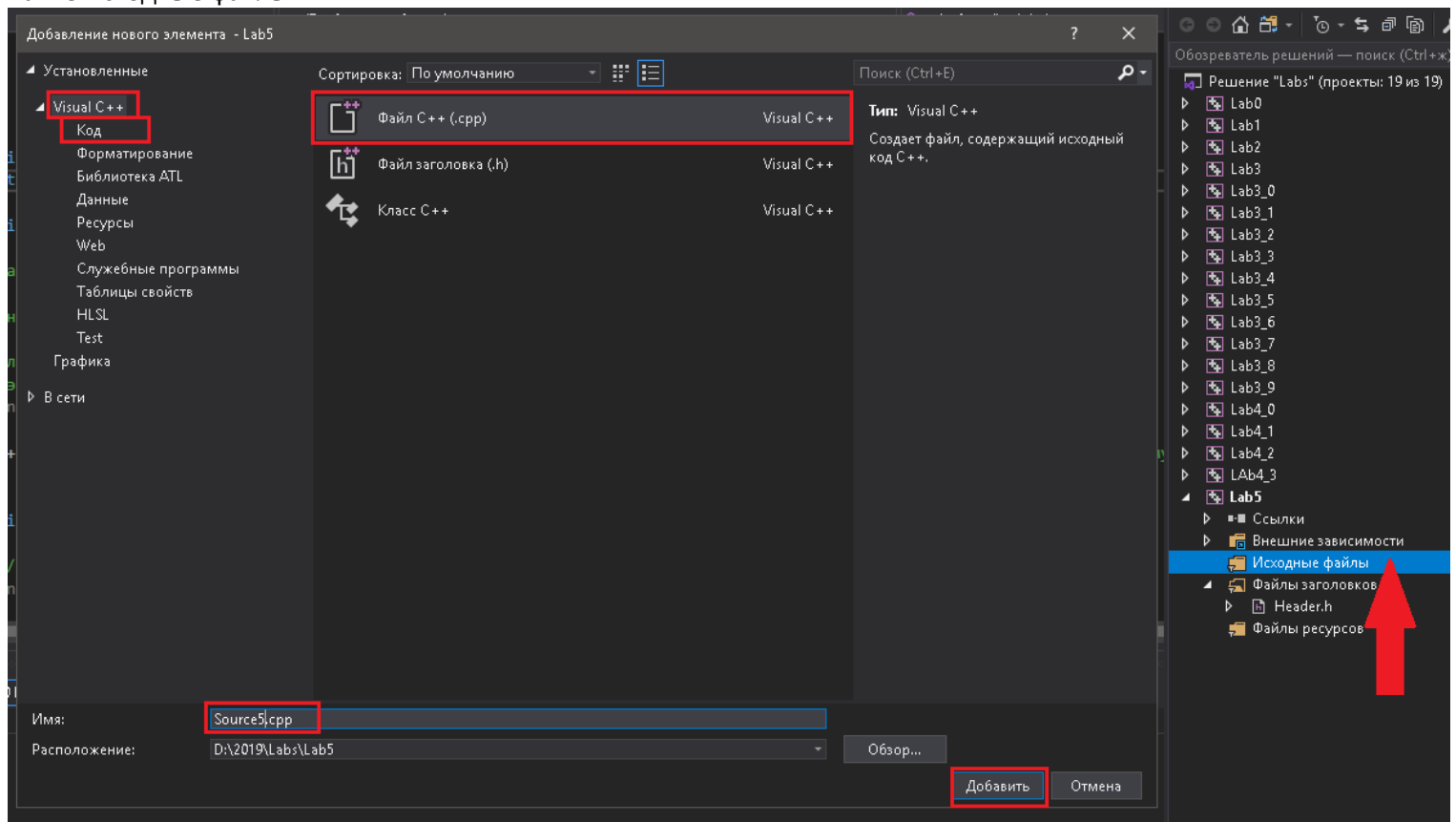
```

Скомпилируем проект, в котором создан только заголовочный файл, кнопкой F5.

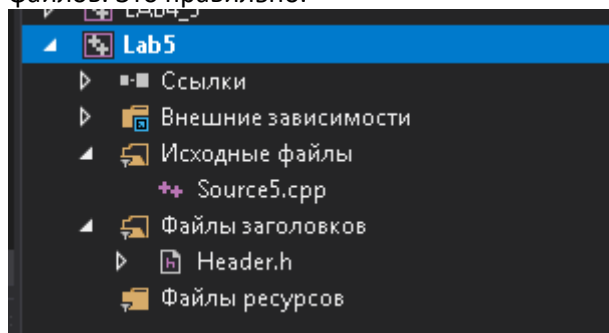


Среда разработки VS2019 сообщает нам, что не может найти исполнимый .exe-файл. Это связано с тем, что в нашем проекте нет .cpp-файла с функцией main(), а компилятор искал именно его и не нашел – соответственно, не смог создать и исполнимый файл .exe на код программы, которую не нашел. Для того, чтобы у нас была программа, нужно создать в нашем проекте исходный .cpp-файл с одной функцией main(). Если исходный файл не будет иметь ошибок, то на его основе можно скомпилировать исполнимый файл.

Создадим в этом же проекте .cpp-файл с функцией main(), в котором подключим наш заголовочный файл, чтобы использовать его и тем самым протестировать. Обычным образом правой кнопкой мыши создаем .cpp-файл в папке Исходные файлы:



В папке Исходных файлов должен отобразиться .cpp-файл. Заголовочный файл находится в папке Заголовочных файлов. Это правильно.



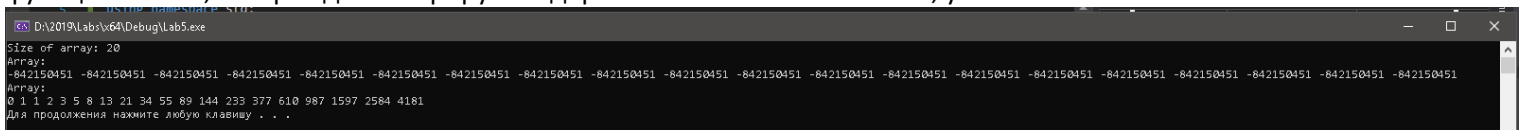
Заполним .cpp-файл кодом программы, которая будет вызывать заголовочный файл и использовать из него функции.

```

1 #include <iostream>
2 // #include "Header.h" // заголовочный файл Header.h находится в папке проекта, где и .cpp-файл, поэтому короткий путь к заголовочному файлу работает
3 // #include "D:\2019\Lab5\Lab5\Header.h" // можно написать полный путь к заголовочному файлу, а если он находится ВНЕ папки с проектом - это сделать обязательно
4 #include "..\..\Lab5\Header.h" // можно написать сокращенный путь к файлу. "..\..\\" означает текущая директория на данном ПК
5 using namespace std;
6
7 int main()
8 {
9     int b;
10    do
11    {
12        cout << "Size of array: ";
13        cin >> b; // пользователь задает размер массива
14    }
15    while (b < 1);
16    int* a = new int[b]; // выделяем память под массив
17    if (a == NULL) // если выделить память не удалось
18    {
19        cout << "No memory.\n"; // сообщаем пользователю
20        system("pause");
21        return 0; // завершаем программу
22    }
23    printArray(a, b); // вызов функции из заголовочного файла. Распечатаем содержимое НЕзаполненного массива
24    fillArrayFibonachi(a, b); // вызов функции из заголовочного файла. Заполним массив числами Фибоначчи
25    printArray(a, b); // вызов функции из заголовочного файла. Распечатаем содержимое заполненного массива
26    delete[] a; // массив уже не нужен - удаляем его
27    a = NULL; // зануляем указатель на него. Сам указатель можно использовать для хранения адреса другого целочисленного одномерного массива или динамической целочисленной переменной
28    system("pause");
29    return 0;
30 }

```

Компилируем и тестируем. Видим, что сначала значения в массиве одинаковы, поскольку он не проинициализирован никакими значениями, это значения по умолчанию, означающие «отсутствующее целое значение». Потом работает функция заполнения массива числами Фибоначчи, после чего повторно вызывается функция печати, которая демонстрирует содержимое элементов массива, успешно заполненного числами Фибоначчи.



```

D:\2019\Lab5\Lab5\Debug\Lab5.exe
Size of array: 20
Array:
-842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451
Array:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
Для продолжения нажмите любую клавишу . . .

```

Поскольку заявлялось, что в заголовочный файл можно помещать переменные и константы, которые становятся глобальными переменными и константами для .cpp-файла, который подключит заголовочный файл с ними, давайте это проверим, создав в нашем заголовочном файле константу и переменную.

Допишем код заголовочного файла:

```

1 #pragma once
2 #include <iostream>
3 #include <cmath>
4 using namespace std;
5
6 void fillArrayFibonachi(int*, int); // прототип функции заполнения массива числами Фибоначчи
7 void printArray(int*, int); // прототип функции печати значений элементов одномерного целочисленного массива на консоль
8 const double myConst = 123.456; // объявим константу
9 int myNumber; // продекларируем переменную, не проинициализировав ее
10
11 void fillArrayFibonachi(int* m, int n) // функция заполнения целочисленного одномерного массива числами Фибоначчи
12 {
13     if (n < 2) // если в массиве меньше 2-х элементов
14     {
15         return; // делать нечего - досрочно завершаем функцию
16     }
17     m[0] = 0; // если выполняется эта строка кода, значит в массиве 2 или больше элементов - их можно заполнять числами Фибоначчи
18     m[1] = 1; // первые 2 элемента массива - это 0 и 1 по умолчанию. Все остальные можно вычислить
19     for (int i = 2; i < n; i++) // проходим по массиву, начиная с элемента с индексом № 2 и далее по всем, заканчивая индексом n-1 включительно
20     {
21         m[i] = m[i - 1] + m[i - 2]; // значение очередного элемента массива есть сумма значений предыдущего элемента и предшествовавшего предыдущему элемента массива
22     } // функция возвращает void - то есть ничего, пустоту. Также можно завершить эту функцию выражением return; // то есть "возвратить ничего"
23 }
24 void printArray(int* m, int n) // функция печати значений элементов одномерного целочисленного массива. Функция ничего не возвращает (void)
25 {
26     cout << "Array:\n"; // напечатаем на консоль предваряющую фразу
27     for (int i = 0; i < n; i++) // цикл для прохода по массиву от первого элемента с индексом № 0 до последнего элемента с индексом n-1
28     {
29         cout << m[i] << " "; // печатаем значение элемента массива с индексом № i и один пробел за ним (пробел - 1 символ, можно взять в одинарные кавычки)
30     }
31     cout << endl; // переведем курсор на новую строку // функция возвращает void - то есть ничего, пустоту. Можно завершить эту функцию выражением return; // то есть "возвратить ничего"
32 }

```

Допишем код .cpp-файла для вызова переменной и константы из заголовочного файла:

```

1 #include <iostream>
2 // #include "Header.h" // заголовочный файл Header.h находится в папке проекта, где и .cpp-файл, поэтому короткий путь к заголовочному файлу работает
3 // #include "D:\2019\Labs\Lab5\Header.h" // можно написать полный путь к заголовочному файлу, а если он находится ВНЕ папки с проектом - это сделать обязательно
4 #include "..\Lab5\Header.h" // можно написать сокращенный путь к файлу. "..\\" означает текущая директория на данном ПК
5 using namespace std;
6
7 int main()
8 {
9     int b;
10    do
11    {
12        cout << "Size of array: ";
13        cin >> b; // пользователь задает размер массива
14    }
15    while (b < 1);
16    int* a = new int[b]; // выделяем память под массив
17    if (a == NULL) // если выделить память не удалось
18    {
19        cout << "No memory.\n"; // сообщаем пользователю
20        system("pause");
21        return 0; // завершаем программу
22    }
23    printArray(a, b); // вызов функции из заголовочного файла. Распечатаем содержимое НЕзаполненного массива
24    fillArrayFibonacci(a, b); // вызов функции из заголовочного файла. Заполним массив числами Фибоначчи
25    printArray(a, b); // вызов функции из заголовочного файла. Распечатаем содержимое заполненного массива
26    delete[] a; // массив уже не нужен - удалим его
27    a = NULL; // зануляем указатель на него. Сам указатель можно использовать для хранения адреса другого целочисленного одномерного массива или динамической целочисленной переменной
28    cout << myConst << endl << myNumber << endl; // печатает значения константы и переменной из заголовочного файла. Они глобальные, а значения глобальных переменных заносятся, то
29    // есть числовым переменным присваивается ноль, char'овской - код ноль (это код непечатного символа). Наша глобальная переменная myNumber по умолчанию инициализируется нулем
30    myNumber = 17; // значение переменной из заголовочного файла можно присвоить новое
31    // myConst = 3.7; // ошибка, константе нельзя присвоить новое значение, пусть даже константа определена в другом файле. Можно только читать значение из константы
32    cout << myNumber << endl; // значение переменной из заголовочного файла должно стать равным 17
33    system("pause");
34    return 0;
35 }

```

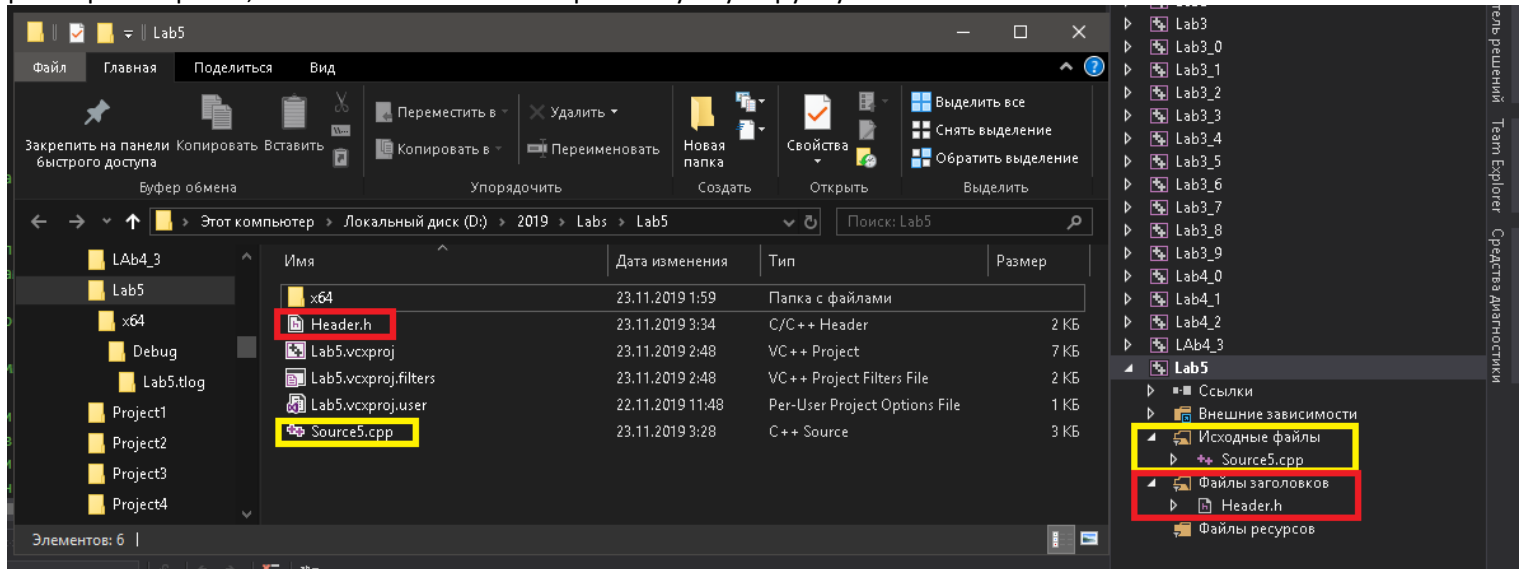
Тестируем. Как видно по результату в консоли, константа доступна для чтения (ее значение распечаталось: 123.456), а НЕпроинициализированная в заголовочном файле переменная, ввиду того, что она является глобальной переменной для программы, автоматически получила значение 0, но поскольку это переменная, то ей можно поменять значение в .cpp-файле, например, на 17.

```

D:\2019\Labs\Lab5\Debug\Lab5.exe
Size of array: 9
Array:
-842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451 -842150451
Array:
0 1 1 2 3 5 8 13 21
123.456
0
17
Для продолжения нажмите любую клавишу . . .

```

Напоследок проверим, какие реально папки находятся в нашем проекте. Кликаем правой кнопкой мыши по имени проекта и в контекстном меню выбираем пункт Открыть папку в проводнике (в «настоящем» проводнике Explorer.exe). В проводнике по папкам и файлам видим, что в нашем проекте реально 5 файлов и папка, которая не отображается в Обзревателе решений. Зато папки, «видимые» в Обзревателе решений на самом деле не существуют, как показывает Обзреватель папок. На самом деле это специальные папки для группировки сходных типов файлов, объединенных общим назначением по группам-«папкам». Но добавлять новые файлы в проект нужно в подходящие папки либо при добавлении кликать мышью по имени самого проекта и в этом случае мы выбираем расширение файла, а VS2019 сама поместит файл в нужную группу.



Для того, чтобы вернуть из функции массив (например, функция его заполняла значениями), надо чтобы функция возвращала указатель на массив. Например, функция func() возвращает указатель на целочисленный одномерный массив:


```
int* func(int* x, int y)
```

```
{
    //code
    return x;
}
```

Например, функция func0() возвращает указатель на вещественный двумерный массив:

```
double** func0(double** ar, int m, int n)
```

```
{
    //code
    return ar;
}
```

В main'е данные функции надо вызвать, передать им указатели на массивы в main'е и присвоить потом указателям на массивы в main'е адрес начала массива, созданного или измененного в функции. Если функция сама выделяет память под массив (а НЕ main), то это сделать **обязательно**. Если память под массив выделяется в main'е, то обычно достаточно передать функции адрес начала массива, а указатель на массив из нее можно и не возвращать. С другой стороны, функция, возвращающая указатель на массив является более функциональной, «продвинутой», а адрес, возвращаемый ею в main'е можно и не отлавливать, если это не надо.

```
int main()
{
    int* vector = NULL; //создается просто указатель на одномерный целочисленный массив. Пусть функция func()
    //выделит под него память
    vector = func(vector, 17); //в таком случае возвращаемый функцией адрес надо отловить и присвоить указателю в
    //main'е.
    double** matrix = NULL; //создается просто указатель на двумерный вещественный массив. Пусть функция func0()
    //выделит под него память
    matrix = func0(matrix, 3, 5);
    //когда массивы не нужны – удалите их
}
```

Задания по вариантам:

В название каждого своего заголовочного файла поместить свою фамилию, например, HeaderAriamnov.h

1	1	Создать двумерный динамический массив типа double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива.
2	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение элементов главной диагонали.
	3	Посчитать в массиве сумму всех элементов, не входящих в главную диагональ.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение квадрата каждого из которых меньше среднего арифметического значения всех элементов массива.
3	1	Создать двумерный динамический массив типа float, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, каждый из которых меньше k (k вводится пользователем).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и больших k (k определяется пользователем).

	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить пользователю индексы первого из них.
	5	Сообщить пользователю количество элементов массива, значение куба каждого из которых больше среднего арифметического значения всех элементов массива.
4	1	Создать двумерный динамический массив типа <code>int</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, значение каждого из которых входит в диапазон от k до p (k и p определяются пользователем).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и значение каждого из которых не входит в диапазон от k до p (k и p определяются пользователем).
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить пользователю индексы последнего из них.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и больше p , определяемого пользователем.
5	1	Создать двумерный динамический массив типа <code>signed long int</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму квадратов значений элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к четным столбцам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, сообщить пользователю индексы первого из них.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива и меньше суммы всех элементов массива.
6	1	Создать двумерный динамический массив типа <code>signed long double</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму корней из значений положительных элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ, каждый из которых меньше суммы элементов главной диагонали.
	4	Найти максимальное и минимальное значение элементов массива. Если они входят в диапазон от x до p (x и p определяются пользователем), сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индексы не принадлежат к главной диагонали.
7	1	Создать двумерный динамический массив типа <code>float</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, каждый из которых больше среднего арифметического значения всех элементов массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не равных x (значение x определяется пользователем).
	4	Найти максимальное и среднее арифметическое значение элементов массива. Если одинаковых максимальных значений в массиве несколько, присвоить каждому из них значение среднего арифметического по изначальному массиву.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива, а индексы не принадлежат к главной диагонали.
8	1	Создать двумерный динамический массив вещественного типа, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов четных столбцов и четных строк.

	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к первой и последней строкам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, присвоить каждому из них значение среднего арифметического изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива, а индекс не относится к нечетным столбцам.
9	1	Создать двумерный динамический массив целочисленного типа, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не относящихся при этом к первой и последней строкам.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и принадлежащих к четным столбцам.
	4	Найти максимальное значение элемента массива. Если в массиве максимальное значение только одно, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индекс не принадлежит к первому и последнему столбцам.
10	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не относящихся к четным строкам.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и больших среднего арифметического значения по всему массиву.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, заменить их минимальным значением массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индекс принадлежит четным строкам.
11	1	Создать двумерный динамический массив типа short int, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение суммы элементов главной диагонали и суммы элементов первой строки массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и при этом меньших суммы значений элементов последнего столбца массива.
	4	Найти максимальное значение элемента массива. Если оно больше минимального значения элемента массива в x раз, сообщить об этом пользователю (значение x вводится пользователем).
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс принадлежит к четным столбцам.
12	1	Создать двумерный динамический массив целочисленного типа, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму корней значений элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к крайним (пограничным) строкам и столбцам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, присвоить им значение максимального элемента массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс относится к четным строкам.
13	1	Создать двумерный динамический массив типа double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения

		динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов побочной диагонали (она противоположна главной диагонали массива).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и четные строки.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, обеспечить пользователю ввод с клавиатуры новых значений для этих элементов.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс не относится к четным столбцам.
14	1	Создать двумерный динамический массив типа float, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение элементов побочной диагонали (она противоположна главной диагонали массива).
	3	Посчитать в массиве сумму всех элементов, не входящих в главную и побочную диагонали.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, заменить все значения массива средним арифметическим значением элементов изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше квадрата среднего арифметического значения всех элементов массива.
15	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не больших среднего арифметического значения всех элементов массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и четные столбцы.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, заменить каждый элемент массива максимальным значением элемента изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше корня из среднего арифметического значения всех элементов массива.
16	1	Создать двумерный динамический массив типа double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива.
17	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение элементов главной диагонали.
	3	Посчитать в массиве сумму всех элементов, не входящих в главную диагональ.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение квадрата каждого из которых меньше среднего арифметического значения всех элементов массива.
18	1	Создать двумерный динамический массив типа float, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом

		и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, каждый из которых меньше k (k вводится пользователем).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и больших k (k определяется пользователем).
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить пользователю индексы первого из них.
	5	Сообщить пользователю количество элементов массива, значение куба каждого из которых больше среднего арифметического значения всех элементов массива.
19	1	Создать двумерный динамический массив типа <code>int</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, значение каждого из которых входит в диапазон от k до p (k и p определяются пользователем).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и значение каждого из которых не входит в диапазон от k до p (k и p определяются пользователем).
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, сообщить пользователю индексы последнего из них.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и больше p , определяемого пользователем.
20	1	Создать двумерный динамический массив типа <code>signed long int</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму квадратов значений элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к четным столбцам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, сообщить пользователю индексы первого из них.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива и меньше суммы всех элементов массива.
21	1	Создать двумерный динамический массив типа <code>signed long double</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму корней из значений положительных элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ, каждый из которых меньше суммы элементов главной диагонали.
	4	Найти максимальное и минимальное значение элементов массива. Если они входят в диапазон от x до p (x и p определяются пользователем), сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индексы не принадлежат к главной диагонали.
22	1	Создать двумерный динамический массив типа <code>float</code> , размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, каждый из которых больше среднего арифметического значения всех элементов массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не равных x (значение x определяется пользователем).
	4	Найти максимальное и среднее арифметическое значение элементов массива. Если одинаковых максимальных значений в массиве несколько, присвоить каждому из них значение среднего арифметического по изначальному массиву.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива, а индексы не принадлежат к главной диагонали.
23	1	Создать двумерный динамический массив вещественного типа, размерность и содержимое элементов

		которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов четных столбцов и четных строк.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к первой и последней строкам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, присвоить каждому из них значение среднего арифметического изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых больше среднего арифметического значения всех элементов массива, а индекс не относится к нечетным столбцам.
24	1	Создать двумерный динамический массив целочисленного типа, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не относящихся при этом к первой и последней строкам.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и принадлежащих к четным столбцам.
	4	Найти максимальное значение элемента массива. Если в массиве максимальное значение только одно, сообщить об этом пользователю.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индекс не принадлежит к первому и последнему столбцам.
25	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не относящихся к четным строкам.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и больших среднего арифметического значения по всему массиву.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, заменить их минимальным значением массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива, а индекс принадлежит четным строкам.
26	1	Создать двумерный динамический массив типа short int, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение суммы элементов главной диагонали и суммы элементов первой строки массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и при этом меньших суммы значений элементов последнего столбца массива.
	4	Найти максимальное значение элемента массива. Если оно больше минимального значения элемента массива в x раз, сообщить об этом пользователю (значение x вводится пользователем).
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс принадлежит к четным столбцам.
27	1	Создать двумерный динамический массив целочисленного типа, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму корней значений элементов главной диагонали.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и не относящихся к крайним (пограничным) строкам и столбцам.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве

		несколько, присвоить им значение максимального элемента массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс относится к четным строкам.
28	1	Создать двумерный динамический массив типа double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов побочной диагонали (она противоположна главной диагонали массива).
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и четные строки.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, обеспечить пользователю ввод с клавиатуры новых значений для этих элементов.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше среднего арифметического значения всех элементов массива и индекс не относится к четным столбцам.
29	1	Создать двумерный динамический массив типа float, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве произведение элементов побочной диагонали (она противоположна главной диагонали массива).
	3	Посчитать в массиве сумму всех элементов, не входящих в главную и побочную диагонали.
	4	Найти максимальное значение элемента массива. Если одинаковых максимальных значений в массиве несколько, заменить все значения массива средним арифметическим значением элементов изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше квадрата среднего арифметического значения всех элементов массива.
30	1	Создать двумерный динамический массив типа long double, размерность и содержимое элементов которого задается пользователем с клавиатуры. Проверить корректность размерности и успешность выделения динамической области памяти под массив. При вводе пользователем ошибочной размерности в цикле сообщить ему об этом и обеспечить ввод пользователем новых значений до тех пор, пока он не введет корректные данные.
	2	Посчитать в массиве сумму элементов главной диагонали, не больших среднего арифметического значения всех элементов массива.
	3	Посчитать в массиве произведение всех элементов, не входящих в главную диагональ и четные столбцы.
	4	Найти минимальное значение элемента массива. Если одинаковых минимальных значений в массиве несколько, заменить каждый элемент массива максимальным значением элемента изначального массива.
	5	Сообщить пользователю количество элементов массива, значение каждого из которых меньше корня из среднего арифметического значения всех элементов массива.