Контрольное домашнее задание, модуль 1

Контрольное домашнее задание предполагает самостоятельную домашнюю работу. Вам потребуется:

1. Изучить предложенные теоретические материалы самостоятельно.
2. Самостоятельно поработать с документацией по языку C#, в т.ч. осуществлять информационный поиск.
3. Разработать программы, определённые основной задачей и индивидуальным вариантом.
4. Сдать заархивированное решение, содержащее два проекта с решениями задач в SmartLMS вовремя.

Формат сдачи работы

Для проверки предоставляется решение, содержащие два проекта консольных приложений. Решение должно быть заархивировано и приложено к заданию в SmartLMS.

Срок выполнения и загрузки работы

Две недели (фактический дедлайн смотреть по SmartLMS)

Дедлайн является мягким и еще на протяжении суток работу можно будет отправить на проверку, с учётом штрафов. Если работа направлена на проверку не более чем через час после дедлайна, то максимальная оценка за нее 8 баллов. Работы, присланные позже, чем через час после дедлайна штрафуются баллом (при максимуме 8) от возможной оценки за каждые 2 часа просрочки.

Основная задача

Обязательно ознакомьтесь с теоретическими материалами для выполнения данной работы.

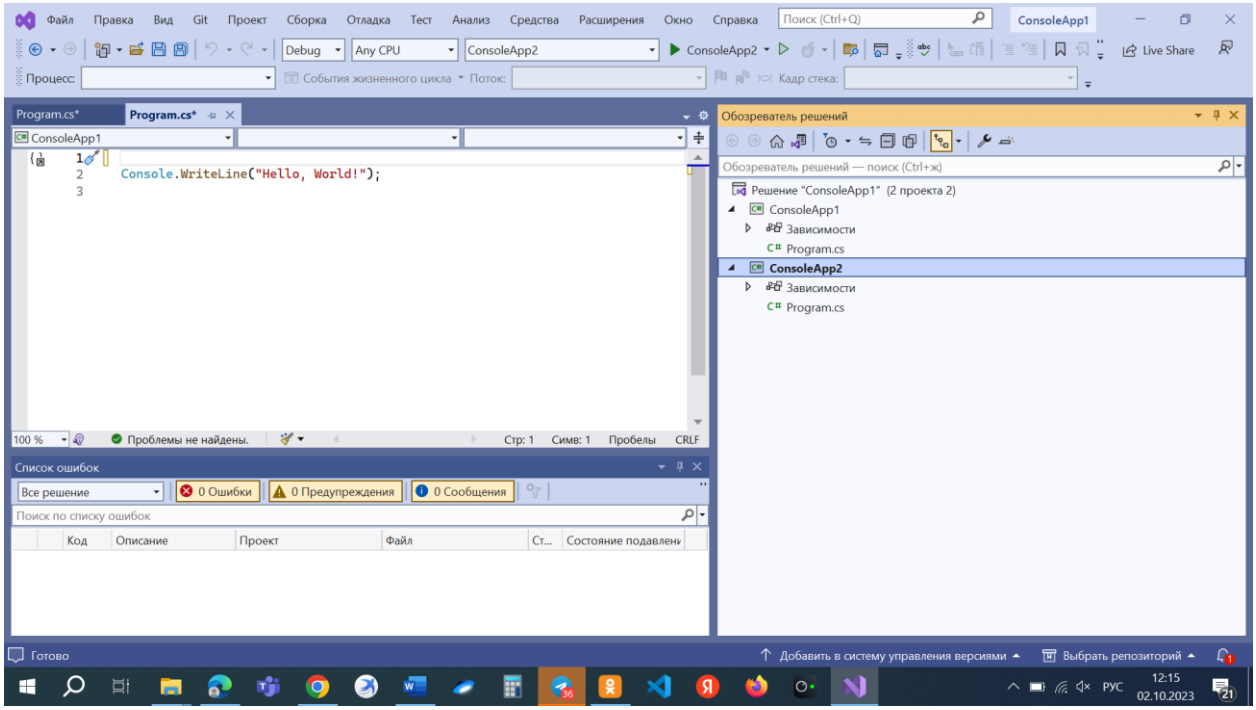
В одном решении разработать два проекта консольных приложений

Для добавления в текущее решение, содержащего Проект 1, нового Проекта 2 выполняем действия (для IDE Visual Studio):

1. В меню Файл выбрать пункт Добавить -> Новый проект и создать его.
2. В файле Programm.cs Проекта 2 разместить код программы.
3. Для запуска на трансляцию исходного модуля Проекта 2 необходимо назначить данный проект текущим (активным). Для этого кликнув на название проекта вызываем правой кнопкой мыши контекстное меню, пункт «Назначить в качестве запускаемого проекта».

В примере проект ConsoleApp2 активен и выделен в обозревателе решений жирным шрифтом.

Обратите внимание, что название решения и проектов должны быть осмысленными, а не автоматически назначаемыми средой разработки.



Первый проект (программа 1):

1. создаёт *структуру данных A*, определённую индивидуальным вариантом (таблица 1, столбец 2);
2. назначает элементам *A* значения, заданные правилом из индивидуального варианта (таблица 1, столбец 3), используя **метод**, в который структура данных A передаётся по правилу, определённому индивидуальным вариантом (таблица 1, столбец 4);
3. создаёт текстовый файл (правило именования файла определено индивидуальным вариантом (таблица 1, столбец 5);
4. размещает в файле, определённом п. 3 требований к программе 1, значения элементов *структуры данных A*, правило форматирования элементов в файле определено индивидуальным вариантом (таблица 1, столбец 5);

Второй проект (программа 2):

1. получает имена файлов для чтения от пользователя и читает файлы формата, определённого требованиями к формату выходных данных программы 1 (таблица 1, столбец 5);
2. если файл не обнаружен на диске / не открывается по произвольным причинам: программа выводит информационное сообщение пользователю (текст строки определите самостоятельно);
3. если нарушен формат данных в файле с данными: программа не извлекает данные и не размещает их в структурах данных программы 2, а выводит информационное сообщение для пользователя (текст строки определите самостоятельно);
4. если файл существует на диске, успешно открылся и содержит данные допустимого формата: программа создаёт *структуру данных B*, определённую индивидуальным вариантом (таблица 2, столбец 2) и загружает в неё элементы из файла;
5. элементы *структуры данных B*, программа обрабатывает по правилу, определённому индивидуальным вариантом (таблица 2, столбец 3), для чего в программе должен быть описан метод обработки, куда структура данных B передаётся по заданному индивидуальным вариантом (таблица 2, столбец 4) правилу, .
6. выводит данные до и после изменений на экран в отформатированном виде, то есть по строкам, разделяя элементы одним пробелом и форматируя с точностью, заданной при создании файла в проекте 1.

Требования по совместимости и качеству для основной задачи:

- весь программный код должен быть написан на языке программирования C# с учётом использования .net 6.0;
- исходный код должен содержать комментарии, объясняющие неочевидные фрагменты и решения, резюме кода, описание целей кода (см. материалы лекции 1);
- использованные в программе идентификаторы должны соответствовать правилам и соглашениям об именовании идентификаторов C# (https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/coding-style/identifier-names);
- представленный к проверке код должен отвечать общим соглашениям о коде C# Microsoft (https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/coding-style/coding-conventions);
- при перемещении папки проекта (копировании / переносе на другое устройство) файлы должны открываться программой также успешно, как и на компьютере создателя, т.е. вне зависимости от путей;
- если загрузка данных происходит из непрямоугольного представления в прямоугольное, то недостающие элементы справа достраиваются нулевыми вещественными значениями;
- программа не допускает пользователя до решения задач, пока вводятся некорректные данные с клавиатуры;
- программа обрабатывает исключительные ситуации, связанные (1) со вводом и преобразованием / приведением данных как с клавиатуры, так и из файлов; (2) с созданием, инициализацией, обращением к элементам массивов и строк.
- обе представленные к проверке программы должны решать все поставленные задачи, успешно компилироваться, не закрываться без пояснений при некорректных вариантах работы и не завешаться аварийно.

Теоретические материалы для самостоятельной работы

Для работы с файлами и проверки существования / возможностей доступа к файлу используйте класс File (https://learn.microsoft.com/ru-ru/dotnet/api/system.io.file?view=net-6.0)

Для записи данных в текстовый файл воспользуйтесь следующими материалами:

- https://learn.microsoft.com/ru-ru/dotnet/api/system.io.file.writealllines?view=net-6.0
- https://learn.microsoft.com/ru-ru/dotnet/api/system.io.file.writealltext?view=net-6.0

Для организации чтения данных из файла воспользуйтесь следующими материалами:

- https://learn.microsoft.com/ru-ru/dotnet/api/system.io.file.readalllines?view=net-6.0

Для разделения строк на подстроки воспользуйтесь методом Split():

- https://learn.microsoft.com/ru-ru/dotnet/csharp/how-to/parse-strings-using-split

		2,00 0,89 0,61 0,48 0,40 0,35 0,32 0,29 0,27		3 2,00 0,89 0,61 0,48 0,40 0,35 0,32 0,29 0,27
4	Массив из N одномерных вещественных массивов	$1 \leq N < 21$ – целое, вводит с клавиатуры пользователь. Длины измерений одномерных массивов, составляющих A – случайные значения из диапазона [1;15). Значения элементов A назначаются подряд (без учета переходов по массивам, составляющим массив) по формуле: $$\frac{\sqrt{n^3+3}}{n^2+5}, n \geq 1 \text{ целое}$$ *Например,* N = 3 *сформированные массивы массивов (пример с округлением данных для удобства представления в задании) могут выглядеть примерно так:* 0,67 0,74 0,78 0,78 0,75 0,72 0,69 0,66 0,63	Метод возвращает тип void, ссылка на A передана в метод по ссылке out; N – параметр метода. (Справочник по C#. Параметры методов - C# | Microsoft Learn)	Имя файла – строка, заданная пользователем с клавиатуры. Место размещение файла – папка с исполнимым файлом консольного приложения 1. При повторных запусках решения на новых входных данных создаётся новый файл для данных; имя запрашивается у пользователя повторно. Если файл уже существует, данные в нем перезаписываются. Данные об одном массиве массивов пишутся по правилу: На первой строке целое число (размерность массива ссылок). Со следующей строки: на каждой строке файла массив, на который указывает очередная ссылка массива ссылок; элементы разделяются ровно двумя символами ?? без пробелов, в конце строки символ ?; числа отформатированы с точностью 3 знака после десятичного разделителя; *Например, для примера из столбца 3:* 3 0,667??0,737??0,782? 0,780??0,754??0,722??0,689? 0,658??0,629?
5	Массив из N вещественных массивов размера M элементов каждый	$0 < N \leq 15;\ 0 < M \leq 10$ – целые, вводит с клавиатуры пользователь Значения элементов A назначаются подряд (без учета переходов по измерениям) по формуле:	Метод возвращает ссылку на структуру A, N и M – параметры метода (Справочник по C#. Параметры методов - C# | Microsoft Learn)	Имя файла – строка, заданная пользователем с клавиатуры. Место размещение файла – папка с исполнимым файлом консольного приложения 1. При повторных запусках решения на новых входных данных создаётся новый файл для данных; имя

		$\frac{2n^2+n-3}{n^3-n^2}, n>1$ целое *Например,* N = 3 и M = 3 *сформированные массивы массивов (пример с округлением данных для удобства представления в задании) могут выглядеть примерно так:* 1,75 1,00 0,69 0,52 0,42 0,35 0,30 0,26 0,23		запрашивается у пользователя повторно. Если файл уже существует, данные в нем перезаписываются. На каждой строке файла подряд сохраняются элементы одномерных массивов, входящих в массив массивов. При этом элементы одномерных массивов разделяются ровно двумя символами ;; без пробелов , в данных об одномерном массиве размещён один символ % без перевода строки; числа отформатированы с точностью 2 знака после десятичного разделителя. *Например, для примера из столбца 3:* 1,75;;1,00;;0,69%0,52;;0,42;;0,35%0,30;;0,26;;0,23%

6	Массив из N вещественных массивов размера M элементов каждый	$0 < N \le 15;\ 0 < M \le 10$ – целые, вводит с клавиатуры пользователь
		Значения элементов A назначаются подряд (без учета переходов по измерениям) по формуле:
		
		$\frac{n^3}{n^2+1}-\frac{3n^2}{3n-1}, n \ge 1$ целое
		
		Например, N = 3 и M = 3
		сформированные массивы массивов (пример с округлением данных для удобства представления в задании) могут выглядеть примерно так:
		`-0,50 -0,40 -0,30`
		`-0,24 -0,19 -0,16`
		`-0,14 -0,12 -0,11`
			Метод возвращает ссылку на структуру A, N и M – параметры метода (Справочник по C#. Параметры методов - C# | Microsoft Learn)	Имя файла – строка, заданная пользователем с клавиатуры.
				Место размещение файла – папка с решением, выше уровня обоих папок проектов.
				При повторных запусках решения на новых входных данных создаётся новый файл для данных; имя запрашивается у пользователя повторно. Если файл уже существует, данные в нем перезаписываются.
				На первой строке целое число (размерность массива ссылок). Со следующей строки: на каждой строке файла массив, на который указывает очередная ссылка массива ссылок; элементы разделяются ровно двумя символами ?? без пробелов, в конце строки единственный символ ?; числа отформатированы с точностью 3 знака после десятичного разделителя;
				Между строками массива массивов пустая строка.
				Например, для примера из столбца 3:
				3
				`-0,500??-0,400??-0,300?`
				
				`-0,235??-0,192??-0,162?`