

Контрольное домашнее задание 1, модуль 2

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 в SmartLMS вовремя заархивированный проект с кодом проекта консольного приложения и библиотеки классов, определённые заданием и вариантом.

Формат сдачи работы

Для проверки предоставляется решение, содержащие два проекта: консольное приложение и библиотеку классов. Решение должно быть заархивировано и приложено в качестве ответа на задание в SmartLMS.

Срок выполнения и загрузки работы

Две недели (фактический дедлайн смотреть по SmartLMS)

Опоздания и штрафы

Дедлайн является мягким и еще на протяжении суток работу можно будет отправить на проверку, с учётом штрафов.

Опоздание в часах	Максимальная оценка, которую можно получить
1	8
2-3	7
4-5	6
6-7	5
8-9	4
9 и более	1

Основная задача

Разработать решение, содержащие проект *консольного приложения* и *библиотеку классов*, включающую статические классы для работы с данными CSV-файла, определённого индивидуальным вариантом (см. таблицу 1).

Консольное приложение

В консольном приложении:

1. Запросить у пользователя абсолютный путь к файлу с **csv**-данными. Допустимый формат данных в файле определён индивидуальным вариантом. Для получения данных из файла использовать метод **Read** класса **CsvProcessing** из библиотеки классов.
2. После получения данных из файла предоставить пользователю экранное меню (пример см. ниже), пункты которого отвечают за манипуляции с файлом и его данными:
 - a. выборка по значениям полей, указанным в индивидуальном варианте (таблица 1, столбец 3). Для организации выборки использовать методы класса **DataProcessing** из библиотеки классов. Конкретные значения полей для организации выборки получать от пользователя. Результат выводить на экран в табличном, читаемом виде. Если результат пуст, то оповещать пользователя.
 - b. сортировка по значениям полей, указанным в индивидуальном варианте (таблица 1, столбец 4). Для организации сортировки использовать методы класса

- DataProcessing** из библиотеки классов. Результат сортировки выводить на экран в табличном, читаемом виде.
- с. сохранение результатов выборок и сортировок в **csv**-файле при помощи перегруженных методов **Write** класса **CsvProcessing** (правила работы с содержимым файла уточнены в разделе «Библиотека классов»). Требуется использовать в работе обе перегрузки: для записи одной строки – одну, для записи массива строк – вторую. Структура файла, т.е. набор полей, должны быть идентичны исходному, включая заголовки. Разрабатываемое консольное приложение должно без ошибок загружать данные из любого созданного ей файла. Имя файла и необходимость сохранения данных запрашивать у пользователя после вывода данных на экран. Сохранять файл рядом с исполнимым файлом консольного приложения. Если пользователь указал некорректное имя файла, то оповещать его и запрашивать повторный ввод.
 3. Реализовать обработку всех исключительных ситуаций, которые могут возникать при работе с вызываемыми методами, включая методы разрабатываемой в рамках КДЗ библиотеки.

Пример экранного меню

Укажите номер пункта меню для запуска действия:

1. Произвести выборку по значению Area
2. Произвести выборку по значению Name
3. Произвести выборку по значению Area и Name
4. Отсортировать таблицу по значению Year (прямой порядок)
5. Отсортировать таблицу по значению Name (прямой порядок)
6. Выйти из программы

Библиотека классов

У библиотеки классов должно быть осмысленное имя. Библиотека содержит статические классы со статическими методами:

1. Класс **CsvProcessing**: содержит методы для чтения и записи данных в csv файл. Для хранения пути к файлу в классе использовать статическое поле **fPath**, к данным этого поля обращаются все методы:
 - Метод **Read** возвращает массив строк (**string[]**) файла, доступного по **fPath**. Если файл отсутствует, или его структура не соответствует варианту, то метод выбрасывает исключение с типом **ArgumentNullException**.
 - Метод **Write** отвечает за запись данных в файл и имеет две перегрузки:
 - Метод с параметрами: строкой (**string**) и путём к новому файлу **nPath**. Метод дописывает (не стирая уже записанные в файл данные) в конец файла данные из параметра-строки после последней строки, если файл по **nPath** уже присутствует на диске. Если файл отсутствует, то он должен быть создан по указанному пути. Файлы, путь до которых указан некорректно, метод не создаёт, продумайте и реализуйте самостоятельно вариант программного оповещения для вызывающего кода.
 - Метод с параметром массивом строк (**string[]**) записывает новые данные, переданные в параметре массиве строк в уже существующий по пути **fPath** файл, стирая его исходное содержимое. Если файл по пути **fPath** отсутствует, то он должен быть создан по указанному пути. Файлы, путь до которых указан некорректно, метод не создаёт, продумайте и реализуйте самостоятельно вариант программного оповещения для вызывающего кода.

Класс **DataProcessing** содержит методы для работы с данными, полученными из файла: **методы получения выборки**, указанных в таблице 1, в столбце 3 и **методы сортировок** по данным полям, указанных в столбце 4. Способ упорядочения также указан в столбце 4 таблицы 1.

Комментарии по работе с данными

В данных есть пропуски и пустые поля. Выводить на экран такие данные не нужно, но их требуется учитывать, если пропуски встречаются при сортировках или фильтрах. Например, размещать в начале или конце списка при сортировке.

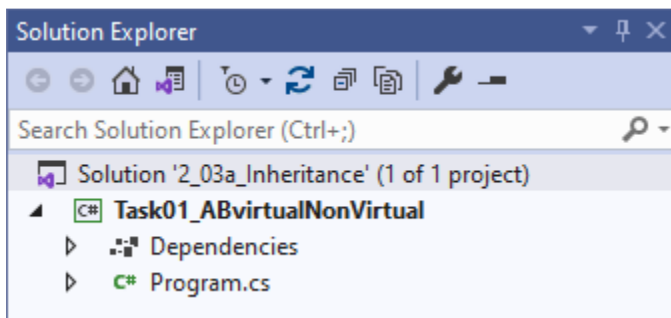
Требования по совместимости и качеству для основной задачи:

- 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](#), модуль 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 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 библиотеки (копировании / переносе на другое устройство) файлы должны открываться программой также успешно, как и на компьютере создателя, т.е. по относительному пути;
- текстовые данные, включая данные на русском языке, успешно декодируются при представлении пользователю и человекочитаемы;
- программа не допускает пользователя до решения задач, пока с клавиатуры не будут введены корректные данные;
- консольное приложение 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; (3) вызовом методов библиотеки.
- представленная к проверке библиотека классов должна решать все поставленные задачи, успешно компилироваться.
- в качестве структур данных использовать только массивы (тип, производный от **Array**).
- для работы с csv-файлами запрещено использовать сторонние библиотеки и nuget-пакеты, код должен быть подготовлен самостоятельно.
- консольное приложение должно обрабатывать аварийные ситуации и содержать цикл повторения решения.
- каждый метод библиотеки может быть использован отдельно от кода консольного приложения, например, при вставке в другой проект и обращении к методам библиотеки из него по ссылке на библиотеку и указанной в задании сигнатуре метода.

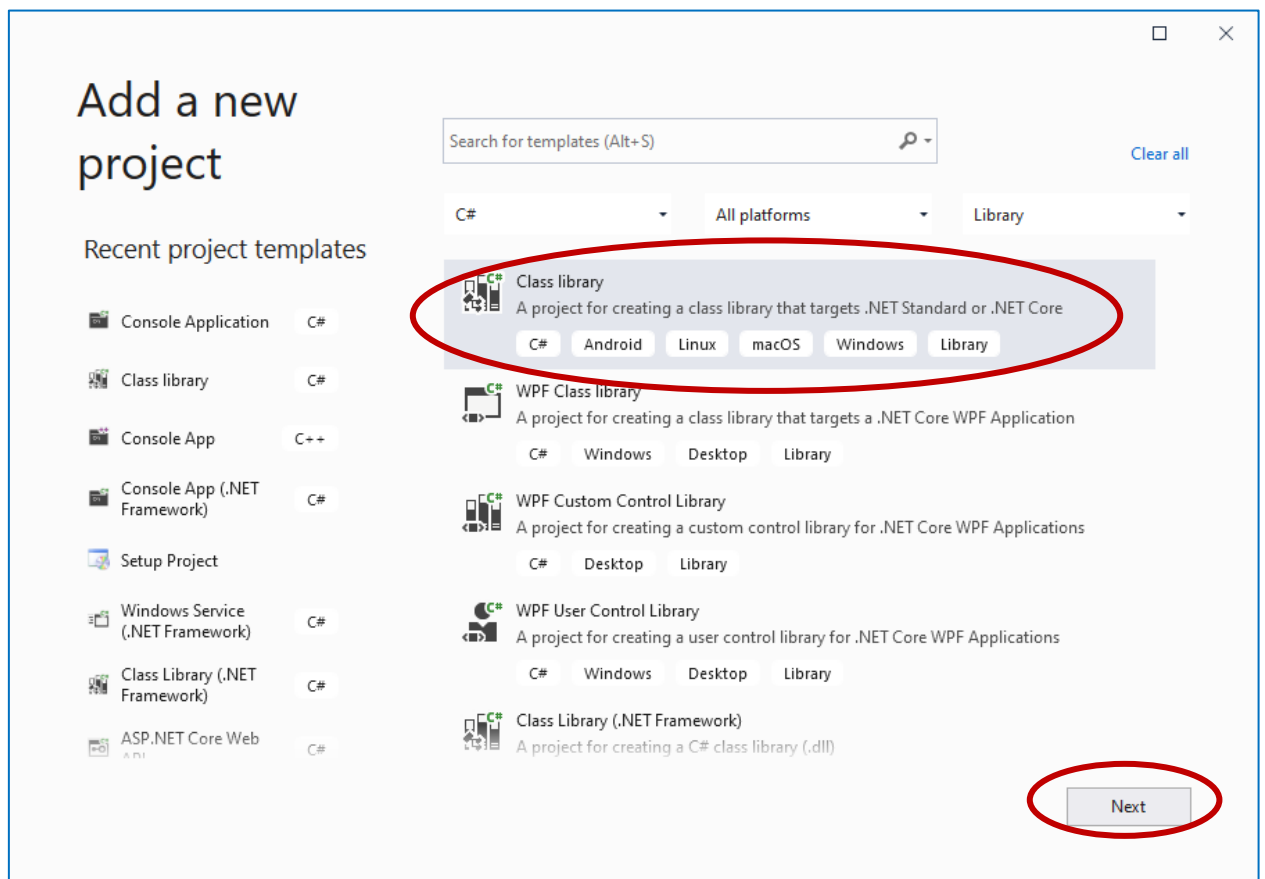
Добавление библиотеки классов в проект

Шаг 1. В решение добавить новый проект Class Library

Для элемента *Решение* в *Обозревателе решений* (Solution Explorer) выбираем контекстное меню для *Решения* и пункт *Добавить новый проект* (Add new project).

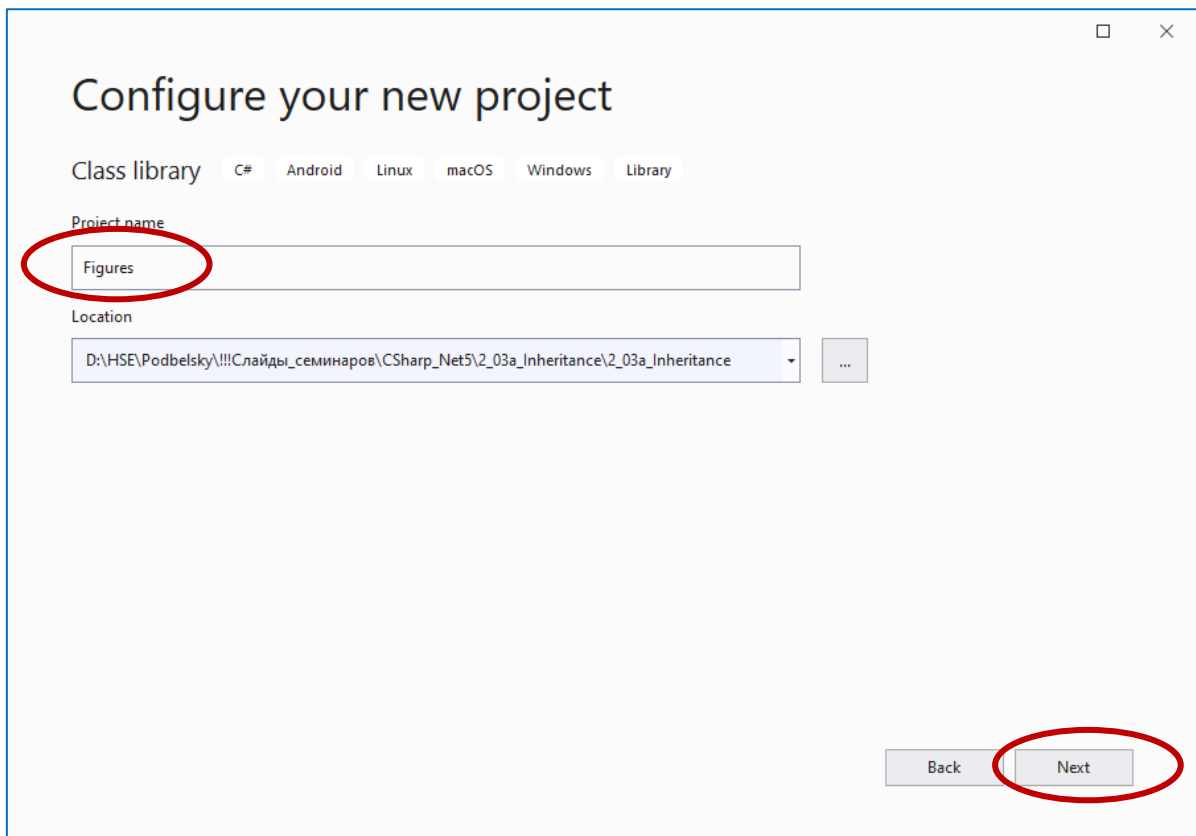


В диалоговом окне добавления нового проекта выбираем шаблон *Библиотека классов*:

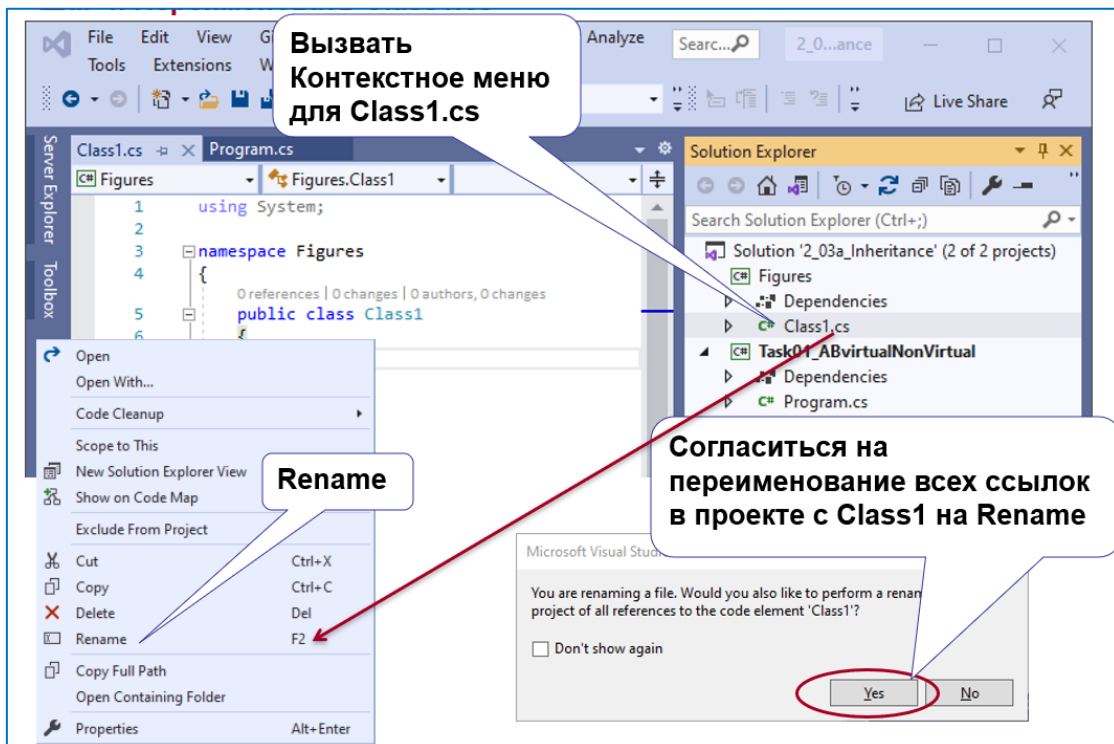


Шаг 2. Задайте имя библиотеки классов

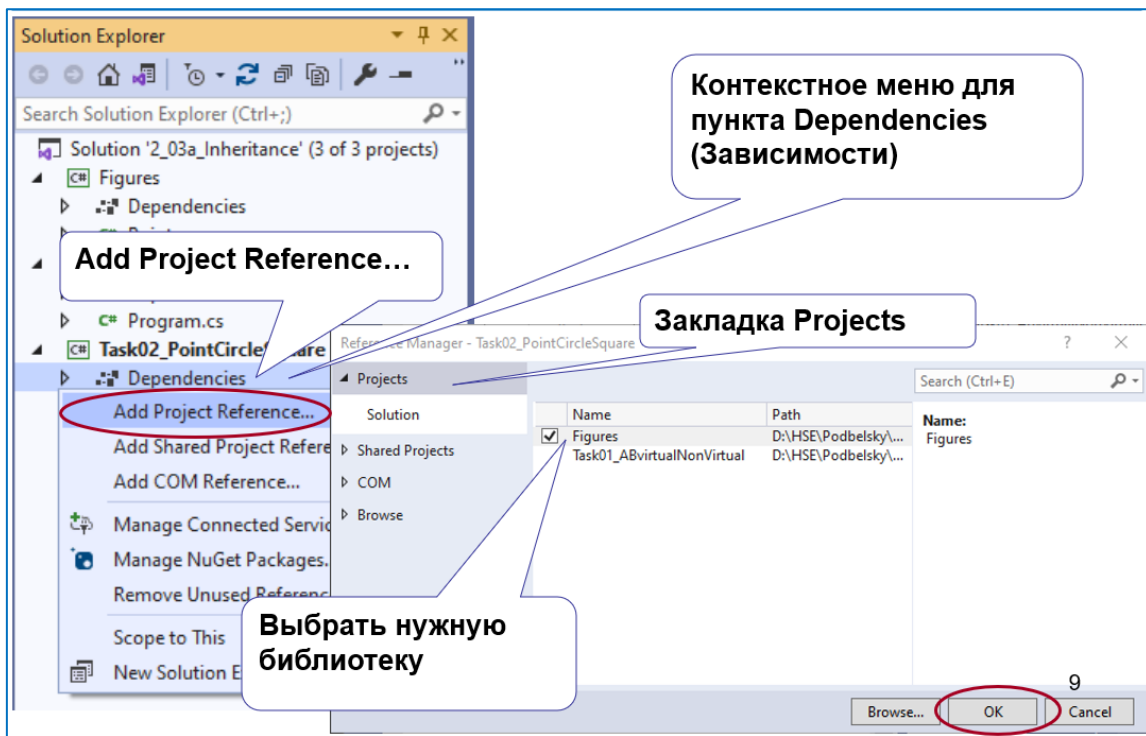
В поле *Название проекта* (Project Name) укажите название библиотеки классов. После нажатия кнопки *Далее* (Next) выбираем платформу .net 6.0.



Шаг 4. Переименовать Class1.cs



Шаг 5. Для подключения библиотеки классов к проекту необходимо в зависимостях проекта добавить ссылку на библиотеку классов.



Индивидуальные варианты

Таблица 1. Индивидуальные варианты

Вариант	Набор данных	Название полей для выборки	Название полей для сортировок
1	2	3	4
1	<i>aeroexpress.csv</i>	<ul style="list-style-type: none"> StationStart StationEnd StationStart и StationEnd 	<ul style="list-style-type: none"> TimeStart в порядке увеличения времени TimeEnd в порядке увеличения времени
2	<i>attraction.csv</i>	<ul style="list-style-type: none"> AdmArea geoarea District и Geoarea 	<ul style="list-style-type: none"> Name по алфавиту в прямом порядке Name по алфавиту в обратном порядке
3	<i>attraction-TC.csv</i>	<ul style="list-style-type: none"> District LocationType AdmArea и Location 	<ul style="list-style-type: none"> AdmArea по алфавиту в прямом порядке AdmArea по алфавиту в обратном порядке
4	<i>botanica-Zaradie.csv</i>	<ul style="list-style-type: none"> LandscapingZone LocationPlace LandscapingZone и ProsperityPeriod 	<ul style="list-style-type: none"> LatinName по алфавиту в прямом порядке LatinName по алфавиту в обратном порядке
5	<i>cult-objects.csv</i>	<ul style="list-style-type: none"> SecurityStatus ObjectType SecurityStatus и Category 	<ul style="list-style-type: none"> ObjectNameOnDoc по алфавиту в прямом порядке ObjectNameOnDoc по алфавиту в обратном порядке
6	<i>electrocar-power.csv</i>	<ul style="list-style-type: none"> AdmArea District AdmArea и пара Longitude_WGS84; Latitude_WGS84 	<ul style="list-style-type: none"> AdmArea по алфавиту в прямом порядке AdmArea по алфавиту в обратном порядке
7	<i>gas-station.csv</i>	<ul style="list-style-type: none"> District Owner AdmArea и Owner 	<ul style="list-style-type: none"> TestDate по возрастанию даты TestDate по убыванию даты
8	<i>geraldic-signs.csv</i>	<ul style="list-style-type: none"> Type RegistrationDate 	<ul style="list-style-type: none"> RegistrationNumber в порядке возрастания номера