



**Тестовые задания по С# (WinForms)**  
**(использовать платформу .NET 4.5 / .NET Core 3.1 или выше)**

1. Подключить библиотеку **Npgsql** через NuGet для работы с базой данных **PostgreSQL** и реализовать класс-прослойку с именем **Query** (IDisposable).  
Реализовать в классе следующие возможности:
  - `bool DeriveParameters()` – получение параметров из запроса
  - `bool OpenConnection()` – открытие соединения с БД
  - `bool CloseConnection()` – закрытие соединения с БД
  - `bool Add(string parameterName, object parameterValue, NpgsqlType type)` – присвоение параметру `parameterName` значения `parameterValue` и типа `type`
  - `bool BeginTransaction()` – начало транзакции
  - `bool CommitTransaction()` – успешное завершение транзакции
  - `bool RollbackTransaction()` – отмена изменений в указанной транзакции
  - `bool ExecuteNonQuery()` – выполнение запроса без получения результата в ответе
  - `DataTable FillData()` – заполнение `DataTable` табличными данными из запроса
2. Реализовать новый класс с наименованием **CustomDataGridView**, который наследуется от стандартного `DataGridView`. Выполнить задания ниже:
  - 2.1. установить по умолчанию режим работы **VirtualMode** = true
  - 2.2. реализовать заполнение данных через `DataSource`, используя в качестве источника данных `DataTable`. Реализовать синхронизацию изменения данных между таблицей и источником, если данные в источнике (`DataTable`) были изменены, они должны сразу измениться в `CustomDataGridView`. Данные `DataTable` получаются из базы (задание 1, метод `FillData()`).
  - 2.3. реализовать фильтрацию данных при условии, что в качестве источника данных выступает `DataTable`. Логика следующая: пользователь нажимает на необходимую колонку в таблице (наименование колонки становится «жирным»), затем если пользователь при активном фокусе таблицы начинает печатать текст на клавиатуре, то выводится форма с `TextBox` (набирается текст для фильтрации) и кнопками «ОК» и «Отмена». После нажатия кнопки «ОК» текст в таблице должен отфильтроваться (через `like %текст%`) на основании выделенной ранее колонки. Пример показан ниже.

ФИО	Телефон	Город	Дата регистрации	Адрес	Почта	Работает	Пользователь	Комментарий
<div style="border: 1px solid gray; padding: 10px; width: fit-content; margin: 0 auto;"> <p>Введите данные</p> <input style="width: 100%;" type="text" value="Иванов"/> <div style="display: flex; justify-content: flex-end; gap: 10px; margin-top: 5px;"> <span>✓ ОК</span> <span>✗ Отмена</span> </div> </div>								

- 2.4. реализовать возможность индивидуального изменения колонок в разрезе пользователя, авторизованного в программе. Пользователь может изменять **положение, ширину и видимость** колонок таблицы с дальнейшим сохранением этих изменений по конкретному авторизованному пользователю (в БД или в файл). При инициализации CustomDataGridView на форме, настройки конкретного пользователя должны автоматически подтянуться и примениться. Идентификация конкретного CustomDataGridView происходит по текстовому уникальному свойству GUID (необходимо добавить в класс новое свойство)
  
3. Реализовать расширяющий метод bool DataCheck() для **ControlCollection**, в котором будет производиться проверка всех элементов управления на форме с интерфейсом ICheckableControl на заполненность данных в них. Если данные заполнены, то метод должен возвращать true
 

```

public interface ICheckableControl
{
    bool EmptyDataCheck { get; set; }
    bool Check();
}
```

Если EmptyDataCheck = true, то через метод Check() для элемента управления происходит проверка значений на пустые данные.
  
4. Реализовать программу (WinForms), на которой будет кнопка для выбора изображения с компьютера (png, jpg, bmp и т. д.). После выбора изображения оно должно открываться на форме для просмотра, где должен быть следующий функционал по работе с изображением: поворачивание, перемещение изображения через зажатие левой кнопкой мыши, уменьшение/увеличение через скролл мыши.
  
5. Реализовать консольную программу на .NET Core, которая будет получать данные со страницы веб-сайта: **наименование, цена со скидкой, цена без скидки, производитель**. Для примера можно использовать любой сайт с аптечными товарами. Список страниц (URL) указывается в настройках (JSON). Полученные данные сохранять в локальную базу данных SQLite. Желательно использовать: библиотеку RestSharp для HTTP запросов, AngleSharp для парсинга HTML.

6. Реализовать класс для отправки сообщений в Discord канал через вебхук с возможностью прикрепления изображения и текстового файла.
7. Реализовать возможность глобального отлавливания всех ошибок приложения (exception), в случае если в коде нету try/catch, чтобы ее обработать. Создать класс **ClientException**, в котором реализовать метод Process() для обработки этих ошибок и отправки оповещений в Discord канал. Сообщение должно содержать следующую информацию:
  - текст ошибки
  - скриншот рабочего стола пользователя (желательно)
  - версия операционной системы
  - имя пользователя в ОС
  - последняя версия NET Framework, установленная на компьютере
8. Используя библиотеку **CefSharp** с NuGet сделать форму, на которой разместить ChromiumWebBrowser. Форма по умолчанию должна открывать страницу с Яндекс.Картами. Реализовать следующие функции для работы с картой:
  - возможность добавление и перемещения меток на карте
  - возможность рисования круга на карте
  - возможность отображения балуна (подсказки) на меткеИспользовать для взаимодействия с картой метод **ExecuteScriptAsync()** из CefSharp.
9. Реализовать класс для преобразования структуры и данных из DataTable в формат JSON. Желательно: использовать библиотеку Newtonsoft.Json.NET, сделать обратное преобразование из JSON в DataTable.
10. Реализовать консольную программу для скачивания всех вложений из писем с почтового ящика с расширением .zip и .rar. Использовать можно любой почтовый сервис.
11. Реализовать атрибут для Enum перечисления с именем **TextAttribute** и текстовым свойством **Name**. Реализовать расширяющий метод для Enum перечисления **ToText()**, который будет возвращать строку, указанную в атрибуте для значения. Пример:

```
public enum NpgsqlType
{
    [Text("anyarray")]
    Array = -2147483648,
    Unknown = 0,
    [Text("int8")]
    Bigint = 1,
    [Text("bool")]
    Boolean = 2,
    [Text("box")]
    Box = 3,
    [Text("bytea")]
    Bytea = 4,
```

```
[Text("circle")]  
Circle = 5
```

```
}
```