Лабораторная работа № 7.

Тема: Windows Forms. Таймеры. Асинхронные операции.

Цель: закрепить теоретические знания и применить их на практике.

Задание:

Вариант №1.

Задание 1. Timer + ProgressBar.
Создайте форму с элементом ProgressBar и кнопками "Старт" и "Стоп". При нажатии кнопки "Старт" ProgressBar начинает заполняться от 0 до 100 с интервалом 500 мс с помощью компонента Timer. Кнопка "Стоп" останавливает процесс.
Подсказки:
Используйте компонент Timer для обновления значения ProgressBar:
timer.Interval = 500; // Интервал в миллисекундах
timer.Tick += Timer_Tick; // Подпишитесь на событие Tick
В обработчике события Tick увеличивайте значение ProgressBar:
if (progressBar.Value < progressBar.Maximum)
    progressBar.Value++;
else
    timer.Stop(); // Остановить таймер при достижении максимума
Для остановки используйте метод timer.Stop().

Задание 2. Timer + PictureBox (Анимация).
Разработайте форму с PictureBox, где изображение движется по кругу с использованием Timer. Добавьте кнопки "Старт" и "Стоп" для управления анимацией.
Подсказки:
Используйте тригонометрические функции для расчета координат:
double angle = 0;
int radius = 100;
int centerX = pictureBox.Width / 2;
int centerY = pictureBox.Height / 2;
int x = (int)(centerX + radius * Math.Cos(angle));
int y = (int)(centerY + radius * Math.Sin(angle));
Обновляйте угол в событии Tick:
angle += 0.1; // Увеличивайте угол для движения

Задание 3. Асинхронная валидация.
Создайте TextBox для ввода email и кнопку "Проверить". При нажатии запустите асинхронную проверку (с задержкой 1 сек), которая в Label выводит "Валиден" или "Ошибка", в зависимости от наличия символа "@".
Подсказка: Используйте async void для обработчика кнопки и регулярные выражения для проверки.

Задание 4. Асинхронные операции + DataGridView
Создайте форму с DataGridView и кнопкой "Загрузить данные". При нажатии кнопки асинхронно загрузите данные из текстового файла или базы данных и отобразите их в DataGridView.
Подсказки:
Используйте метод File.ReadAllLinesAsync для чтения файла:
private async void LoadDataButton_Click(object sender, EventArgs e)
{
    string[] lines = await File.ReadAllLinesAsync("data.txt");
    dataGridView.DataSource = lines.Select(line => new { Data = line }).ToList();

```
}
```
Обеспечьте блокировку кнопки во время загрузки, чтобы избежать повторных запросов.

Задание 5. Timer + ListBox (Обновление данных).
Создайте форму с ListBox и Timer. Каждые 2 секунды добавляйте новую строку в ListBox с текущим временем. Добавьте кнопку "Очистить", чтобы очистить ListBox.
Подсказки:
Используйте DateTime.Now для получения текущего времени:
```
listBox.Items.Add(DateTime.Now.ToString("HH:mm:ss"));
```
Для очистки используйте метод listBox.Items.Clear().

Вариант 2
Задание 1. Timer + Label (Обратный отсчет).
Создайте форму с Label и кнопками "Старт" и "Стоп". При нажатии кнопки "Старт" начните обратный отсчет от 60 до 0 с интервалом 1 секунда. Отображайте текущее значение в Label.
Подсказки:
Используйте переменную для хранения текущего значения:
```
int countdown = 60;
timer.Interval = 1000; // Интервал 1 секунда
```
В обработчике события Tick уменьшайте значение:
```
if (countdown > 0)
{
    label.Text = countdown.ToString();
    countdown--;
}
else
{
    timer.Stop();
}
```

Задание 2. Асинхронные операции + PictureBox.
Создайте форму с PictureBox и кнопкой "Загрузить изображение". При нажатии кнопки асинхронно загрузите изображение из интернета и отобразите его в PictureBox.
Подсказки:
Используйте класс HttpClient для загрузки изображения:
```
private async void LoadImageButton_Click(object sender, EventArgs e)
{
    using (HttpClient client = new HttpClient())
    {
        byte[] imageBytes = await client.GetByteArrayAsync("https://example.com/image.jpg");
        pictureBox.Image = Image.FromStream(new MemoryStream(imageBytes));
    }
}
```

Задание 3. Асинхронное программирование + ProgressBar.
Создайте форму с ProgressBar и кнопкой "Запустить задачу". При нажатии кнопки запустите асинхронную задачу, которая имитирует длительную операцию (например, загрузку данных). ProgressBar должен показывать прогресс выполнения задачи.
Подсказки:
Используйте ключевое слово async и метод Task.Delay для имитации задержки:
```
private async void StartTaskButton_Click(object sender, EventArgs e)
{
    for (int i = 0; i <= 100; i++)
    {
```

```
        progressBar.Value = i;
        await Task.Delay(50); // Задержка 50 мс
    }
}
```
Убедитесь, что UI остается отзывчивым во время выполнения задачи.

Задание 4. Асинхронные операции + TextBox.
Создайте форму с TextBox и кнопкой "Проверить текст". При нажатии кнопки асинхронно проверьте, является ли введенный текст палиндромом. Результат выведите в метку.
Подсказки:
Используйте метод Task.Run для выполнения проверки в фоновом потоке:
```
private async void CheckTextButton_Click(object sender, EventArgs e)
{
    string text = textBox.Text;
    bool isPalindrome = await Task.Run(() => IsPalindrome(text));
    label.Text = isPalindrome ? "Это палиндром" : "Не палиндром";
}

private bool IsPalindrome(string text)
{
    string reversed = new string(text.Reverse().ToArray());
    return text.Equals(reversed, StringComparison.OrdinalIgnoreCase);
}
```

Задание 5. Timer + DataGridView (Обновление таблицы).
Создайте форму с DataGridView и Timer. Каждые 5 секунд добавляйте новую строку в таблицу с текущим временем и случайным числом.
Подсказки:
Используйте класс Random для генерации чисел:
```
Random random = new Random();
dataGridView.Rows.Add(DateTime.Now.ToString("HH:mm:ss"), random.Next(1, 100));
```
Убедитесь, что таблица автоматически обновляется.

Общие подсказки:
Timer: Используйте свойство Interval для установки временного интервала.
Подпишитесь на событие Tick для выполнения действий.
Методы Start() и Stop() управляют работой таймера.
Асинхронное программирование:
Используйте ключевые слова async и await.
Метод Task.Delay помогает имитировать задержки.
Не забывайте блокировать элементы управления во время выполнения асинхронных операций.

Отчет должен содержать (см. образец):
- номер и тему лабораторной работы;
- фамилию, номер группы студента и вариант задания;
- скриншоты окна Visual Studio с исходным кодом программ и комментариями;
- скриншоты с результатами выполнения программ;
- пояснения, если необходимо;
- выводы.

Отчеты в формате **pdf** отправлять на email: **colledge20education23@gmail.com**