**Практическая работа: Отладка проекта**

**Цель работы**

Закрепление практических навыков работы с Visual Studio 2019 и использование инструментальных средств для проведения отладки приложений.

**Ход работы**

**1. Разработка программы**  
Для демонстрации работы отладчика создадим программу на языке C#, которая вычисляет факториал числа с использованием рекурсии.

**Код программы:**

csharp

Копировать код

using System;

namespace DebuggingExample

{

class Program

{

static void Main(string[] args)

{

Console.Write("Введите число для вычисления факториала: ");

int number;

if (int.TryParse(Console.ReadLine(), out number))

{

Console.WriteLine($"Факториал числа {number} равен {CalculateFactorial(number)}");

}

else

{

Console.WriteLine("Ошибка: введено некорректное число.");

}

Console.ReadKey();

}

static int CalculateFactorial(int n)

{

if (n < 0)

throw new ArgumentException("Число должно быть неотрицательным.");

if (n == 0 || n == 1)

return 1;

return n \* CalculateFactorial(n - 1);

}

}

}

**2. Режим отладки**  
Переходим в режим отладки с использованием инструментов Visual Studio 2019.

1. **Установка контрольных точек:**
   * Установите точки останова на следующих строках кода:
     + На строке int number;.
     + На строке Console.WriteLine($"Факториал числа {number} равен {CalculateFactorial(number)}");.
     + В теле метода CalculateFactorial, на строках:
       - if (n < 0) throw new ArgumentException(...);
       - return n \* CalculateFactorial(n - 1);.

Установка выполняется нажатием **F9** или кликом на левом поле редактора рядом с номером строки.

1. **Запуск программы:**  
   Нажмите **F5** для запуска программы в режиме отладки.
2. **Остановка на точках останова:**  
   Программа останавливается на первой точке останова. Откройте окно **Autos**, чтобы увидеть значения переменных.
3. **Изменение значений переменных:**
   * В момент остановки выполнения программы измените значение переменной number в окне **Locals**. Например, задайте вместо введенного пользователем значения 5.
4. **Пошаговое выполнение программы:**  
   Используйте клавиши:
   * **F10** для выполнения текущей строки.
   * **F11** для перехода внутрь метода.
5. **Просмотр исключений:**
   * Введите отрицательное число при запуске программы, чтобы вызвать исключение. Убедитесь, что программа останавливается на операторе throw.

**3. Просмотр переменных:**

* Используйте окно **Autos** для анализа переменных, к которым недавно обращались.
* Разверните содержимое объекта в окне **Locals**, если работаете со сложным типом данных (например, классом).

**4. Использование условных точек останова:**  
Создайте точку останова с условием:

* Щелкните правой кнопкой мыши по точке останова → выберите **Conditions**.
* Укажите условие n == 3 для строки return n \* CalculateFactorial(n - 1);.

Программа остановится только при достижении этой точки с указанным значением переменной.

**5. Продолжение отладки:**  
Используйте меню **Debug** → **Step Into**, **Step Over** и **Step Out**, чтобы выйти из текущего метода и вернуться в основной поток выполнения.

**Вывод**

В ходе работы были изучены основные инструменты отладки Visual Studio 2019, включая:

* Установку точек останова (обычных и условных).
* Пошаговое выполнение программы.
* Использование окон **Autos**, **Locals**, **Watch** для анализа переменных.
* Обработку и анализ исключений.

Навыки работы с отладчиком позволяют быстрее находить ошибки в коде, исследовать его поведение и вносить изменения для достижения корректной работы программы.

**Ответы на контрольные вопросы**

1. **Что такое отладка?**  
   Отладка — это процесс поиска и устранения ошибок (или багов) в коде программы с использованием специальных инструментов, таких как отладчики.
2. **Какие инструменты отладки вам известны?**
   * **Точки останова (breakpoints).**
   * Окна **Autos**, **Locals**, **Watch**.
   * Исключения и их обработка.
   * Пошаговое выполнение программы (Step Into, Step Over).
   * Логирование.
3. **Методы отладки:**
   * Использование отладчика в IDE (например, Visual Studio).
   * Логирование (запись данных в журнал для последующего анализа).
   * Метод "печати" (вывод значений переменных в консоль).
   * Использование тестов для проверки функциональности.

4o