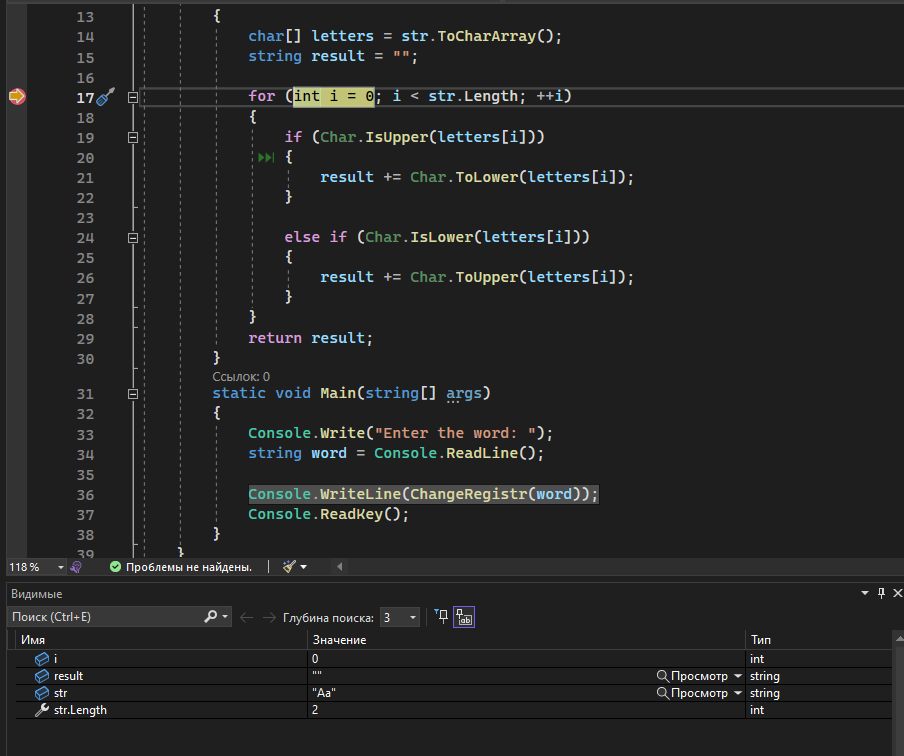
# ПЗ 20. Изучение отладчика интегрированной среды разработчика

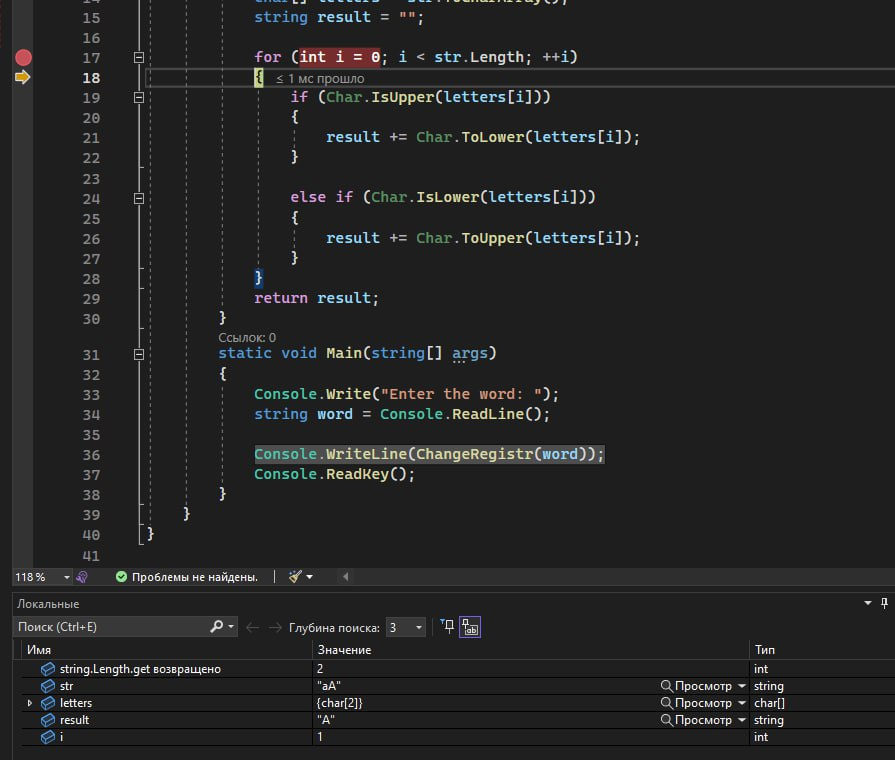
**Задание:** *(отмеченные пункты работы сопровождать скриншотами с пояснениями в отчете)*

## Сформируйте отчет по следующим пунктам работы:

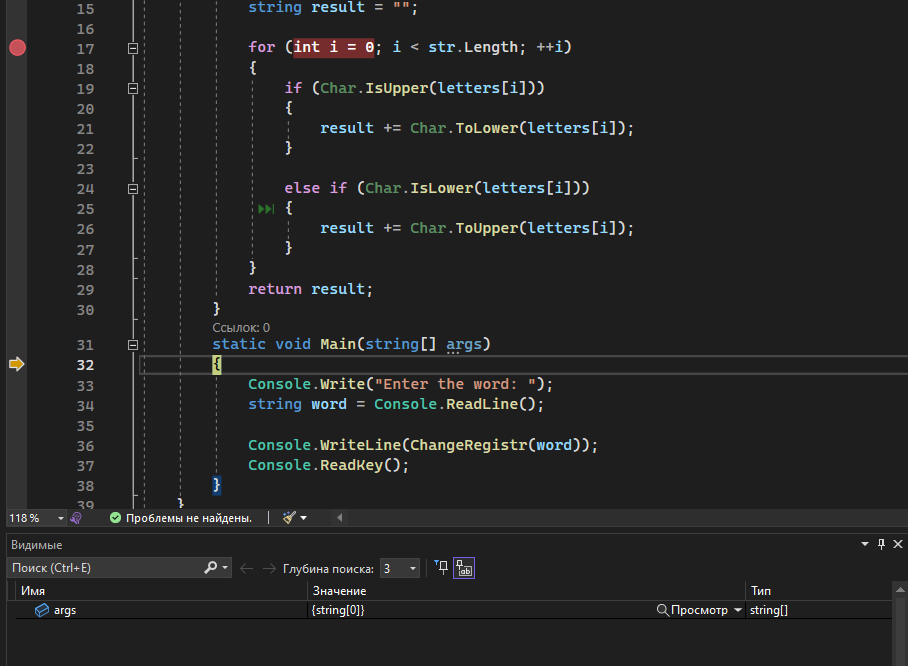
1. Создайте шаблон приложения, следуя инструкциям из п 1 дополнительных сведений.
2. Используя ранее созданный шаблон (п 2. дополнительных сведений), создайте приложение (консольное или WPF на выбор) со следующей функциональностью:
   1. Метод принимает на вход произвольную строку
   2. Метод на выход подает строку с измененным регистром каждого символа ( Hello - > hELLO)
   3. Обязательно использовать циклы, методы для работы с символами и любые вспомогательные конструкции по желанию
3. Задайте значок приложения в свойствах проекта
4. Произведите отладку кода с использованием точек останова:
   1. Установите точку останова внутри метода на строке итерации цикла. Запустите стандартную отладку (F5). Проверьте содержимое счетчика цикла.



* 1. Установите точку останова на строке итерации цикла. Запустите отладку (Шаг с заходом), проверьте содержимое счетчика цикла.

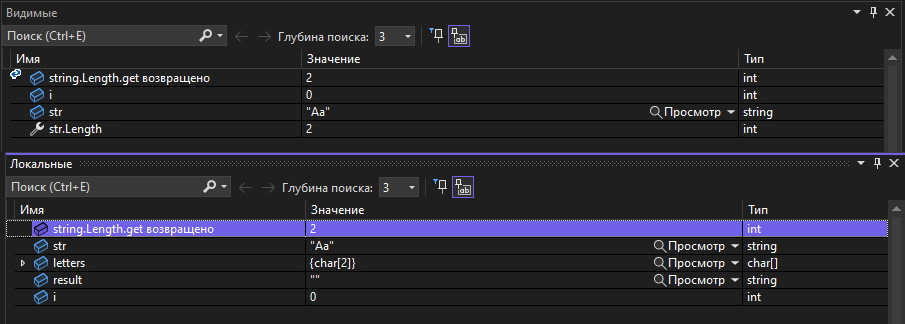


* 1. Сделайте **выводы** по 2-м последним пунктам.  
     Вывод: Мы поставили точку останова на строке с итерацией, и, запустив программу в режиме отладки, программа выполнилась до точки останова. Каждый раз, нажимая клавишу F11 (Шаг с заходом), мы построчно выполняем строку за сторокой. В это время мы можем наблюдать изменение переменных, которое происходит во время исполнения кода.
  2. Используйте дополнительные свойства отладки (на выбор) для наблюдения за поведением кода



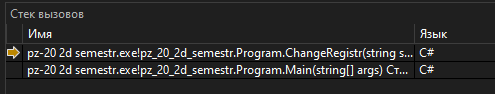
(шаг с обходом)

1. Проверьте изменения значений переменных при отладке в окне Видимые и Локальные (п.5), сделайте **выводы**

****

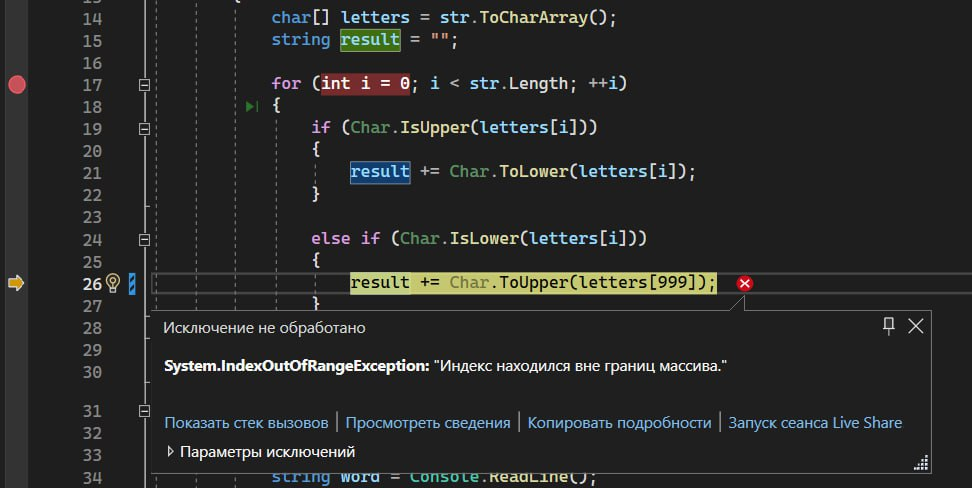
Вывод: Во время исполнения кода, мы видим изменения переменных. Во вкладке Видимые отображены все переменные в целом, а во вкладке Локальные - только переменные в данной области видимости. Это позволяет, например, отслеживать работу алгоритмов.

1. Пронаблюдайте за состоянием стека вызовов при отладке и сделайте **вывод**.



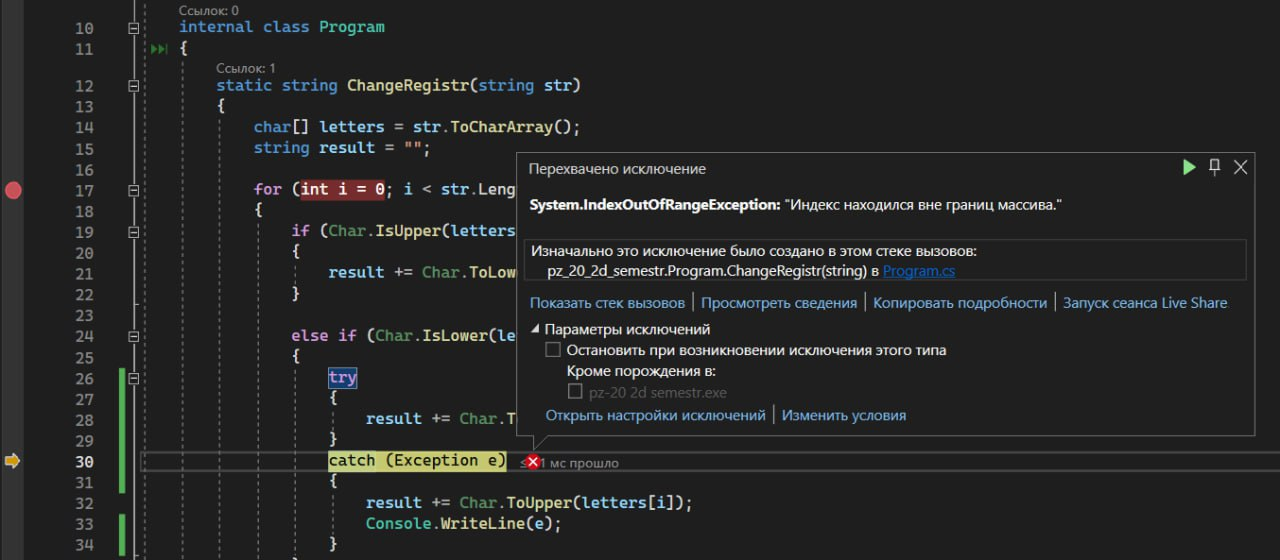
Вывод: с помощью окна «Стек вызовов» можно просматривать вызовы функций и процедур, которые в данный момент находятся в стеке. Желтая стрелка указывает га строку, на которой находится указатель выполнения. Это хорошо подходит для изучения и анализа потока выполнения приложения.

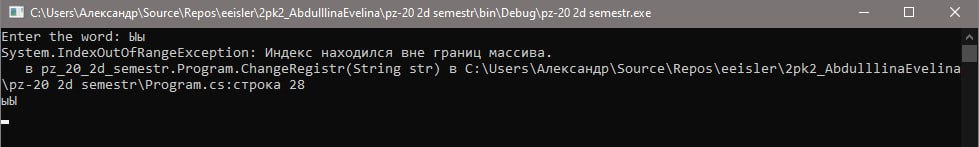
Добавьте в проект код, вызывающий исключение (например выход за пределы строки) проанализируйте сообщение об исключении (п.4) – **вывод** оформить в отчете



Вывод: Мы специально добавили код, вызывающий ошибку. В режиме отладки, дойдя до строки с ошибкой, мы увидели ее сообщение, которое гласит: «Исключение не обработано. System.IndexOutOfRangeException: Индекс находится вне границ массива». Если бы ошибка была добавлена не предусмотрительно, то найти в чем проблема было бы тяжело. Однако, используя отладчик, сделать это легко.

1. Добавьте обработку исключения и приложите скриншот результата с **выводом**.





Вывод: Мы добавили обработку исключения, используя конструкцию try-catch. Теперь в сообщении мы видим, что исключение перехвачено, видим в каком стеке вызовов было создано это исключение.