Шаг 1:

1. Открыли проект в Visual Studio
2. Расположили точки остановы:

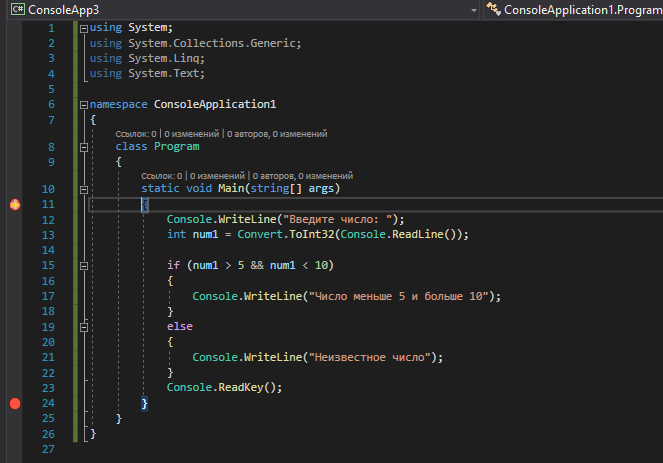


Рис.1

1. Открыли окно средства диагностики:

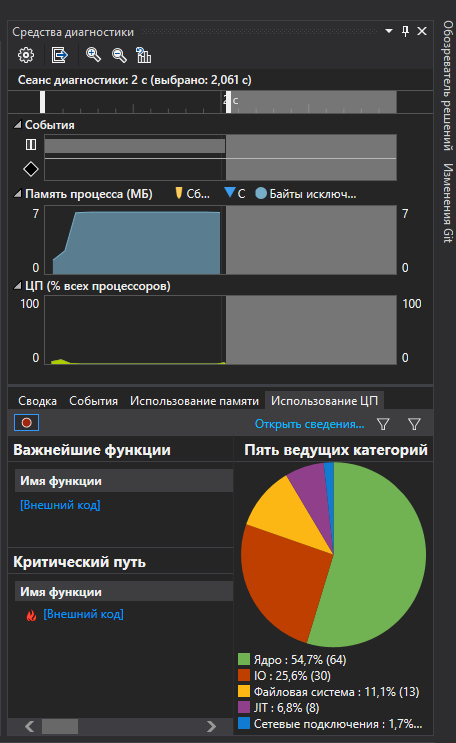


Рис.2

1. Убедились,что включен параметр загрузка ЦП:

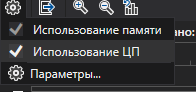


Рис.3

1. Результат после отладки:

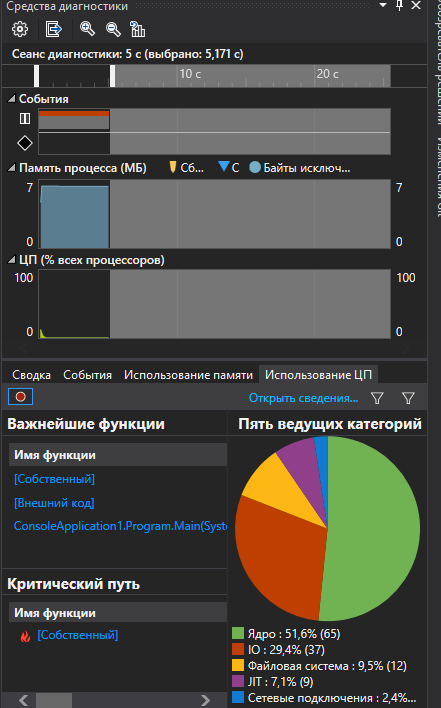


Рис.4

1. Результат после запуска сценария, который вызвал срабатывание первой точки остановки:

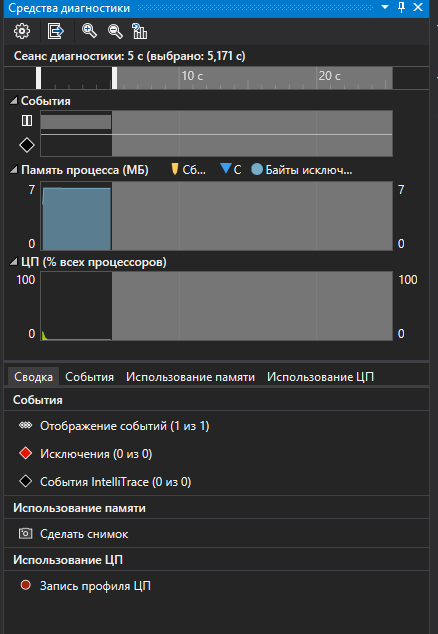


Рис.5

1. Приостановив отладчик, включите сбор данных о загрузке ЦП, а затем откройте вкладку **Загрузка ЦП**
2. Нажмите клавишу F5, чтобы запустить приложение до второй точки останова.

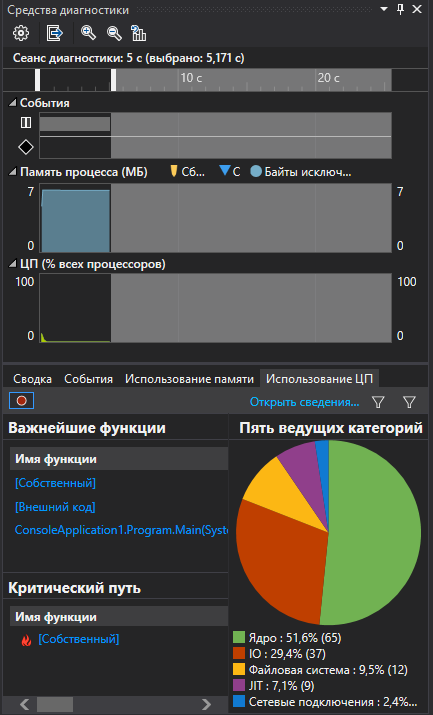


Рис.6

1 точка

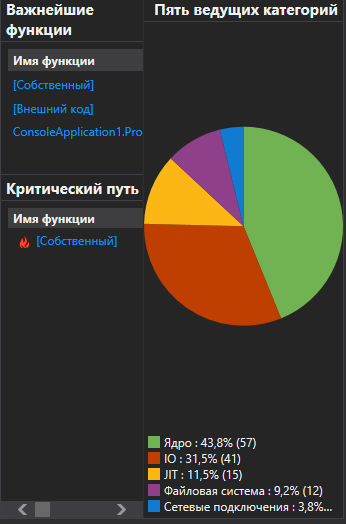


Рис.7

2 точка

1. Если вы хотите выбрать для анализа более конкретную область кода, выберите область на временной шкале ЦП (это должна быть область, в которой отображаются данные профилирования).

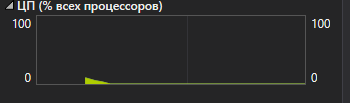


Рис.8

Шаг 2. Анализ данных о загрузке ЦП

1. В списке функций дважды щелкните одну из функций вашего приложения, которая выполняет много работы.

При двойном щелчке функции в левой панели откроется представление **Вызывающий/вызываемый**.

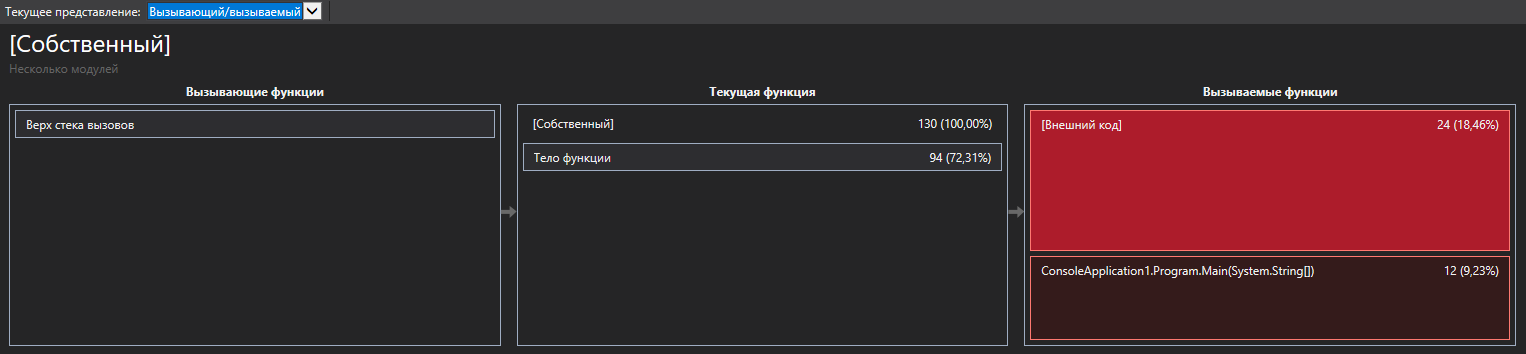


Рис.8

1. Чтобы увидеть более обобщенное представление, показывающее порядок, в котором вызываются функции, выберите в раскрывающемся списке в верхней части панели пункт **Дерево вызовов**.

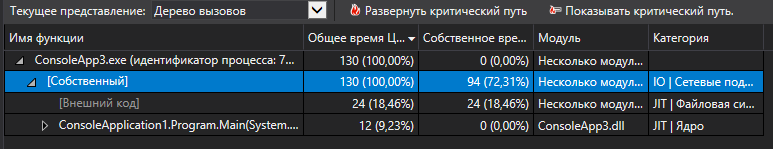


Рис.9

Чтобы увидеть вызовы функций, которые используют самый высокий процент ЦП в представлении дерева вызовов, нажмите **Развернуть критический путь**.

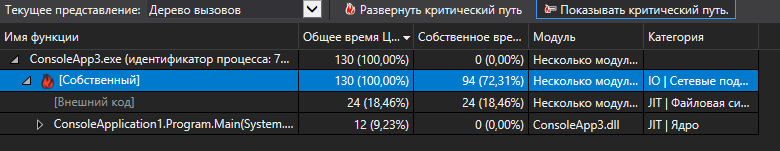


Рис.10

### Просмотр внешнего кода

### 

### Рис.11

Репозиторий проекта: https://github.com/chahzhik/aue