Артем Несин

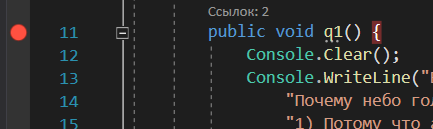
ИС-31

Практика

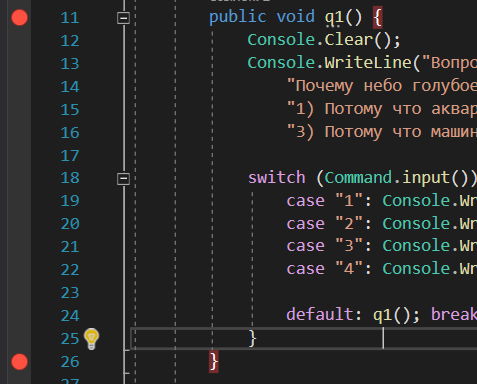
Github - https://github.com/inesin1/mayerPractic1

Шаг 1. Сбор данных профилирования

1. Откройте проект для отладки в Visual Studio и установите точку останова в приложении в точке, где вы хотите проверить загрузку ЦП.

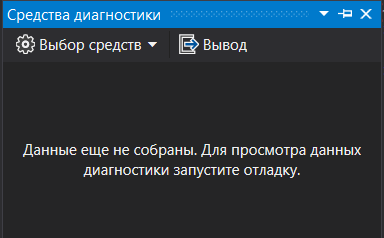


1. Установите вторую точку останова в конце функции или области кода, который требуется проанализировать.

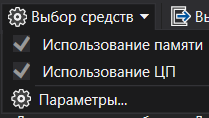


С помощью двух точек останова можно ограничить сбор данных частями кода, которые требуется проанализировать.

1. Окно **Средства диагностики** появится автоматически, если вы не отключали эту функцию. Чтобы снова открыть окно, щелкните **Отладка** > **Окна** > **Показать средства диагностики**.



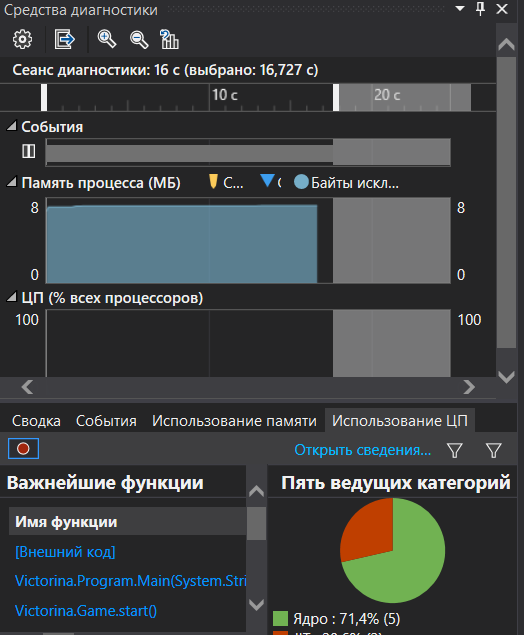
1. Вы можете выбрать, что следует просмотреть, [Использование памяти](https://docs.microsoft.com/ru-ru/visualstudio/profiling/memory-usage?view=vs-2022) или **Загрузка ЦП** (либо оба средства), с помощью параметра **Выбор средств** на панели инструментов. В Visual Studio Enterprise также можно включить или отключить IntelliTrace, выбрав **Сервис** > **Параметры** > **IntelliTrace**.



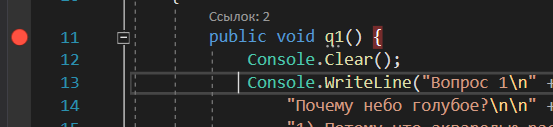
Нас главным образом интересует загрузка ЦП, поэтому убедитесь, что средство **Загрузка ЦП** включено (оно включено по умолчанию).

1. Щелкните **Отладка** > **Начать отладку** (**Запустить** на панели инструментов или **F5**).

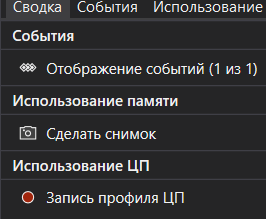
По завершении загрузки приложения отображается представление "Сводка" средств диагностики. Если вам нужно открыть окно, щелкните **Отладка** > **Окна** > **Показать средства диагностики**.



1. Запустите сценарий, который вызвал срабатывание первой точки останова.



1. Приостановив отладчик, включите сбор данных о загрузке ЦП, а затем откройте вкладку **Загрузка ЦП**.

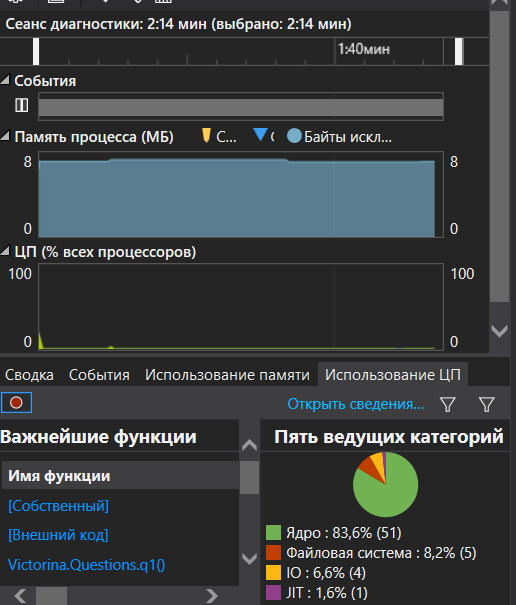


При выборе пункта **Запись профиля ЦП** Visual Studio начнет записывать функции и сведения о времени их выполнения. Эти собранные данные можно просматривать только в том случае, если приложение останавливается в точке останова.

1. Нажмите клавишу F5, чтобы запустить приложение до второй точки останова.

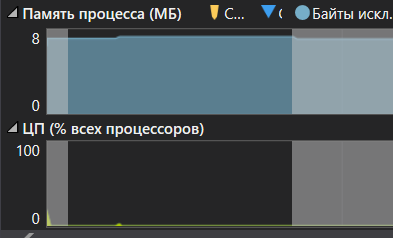
Теперь у вас есть данные о производительности приложения именно для той области кода, которая выполняется между двумя точками останова.

Профилировщик начинает подготавливать данные потока. Дождитесь завершения этой операции.



Средство "Загрузка ЦП" выведет отчет на вкладке **Загрузка ЦП**.

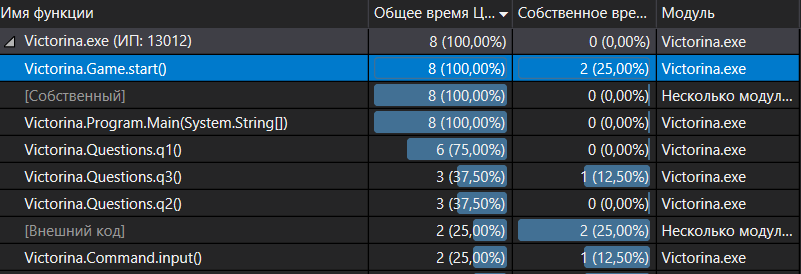
1. Если вы хотите выбрать для анализа более конкретную область кода, выберите область на временной шкале ЦП (это должна быть область, в которой отображаются данные профилирования).



### Шаг 2. Анализ данных о загрузке ЦП

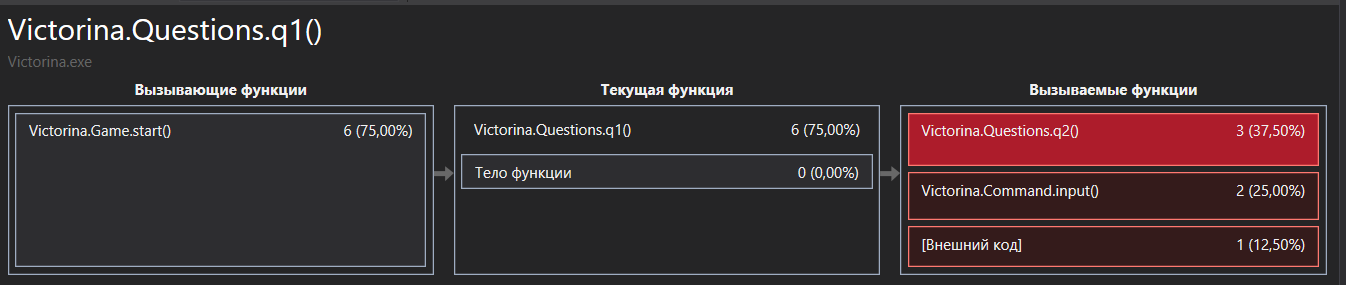
Мы рекомендуем начать анализ данных с проверки списка функций на вкладке "Загрузка ЦП" и выявления функций, выполняющих основную часть работы, а затем подробно рассмотреть каждую из этих функций.

1. В списке функций изучите функции, которые выполняют большую часть работы.



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**.

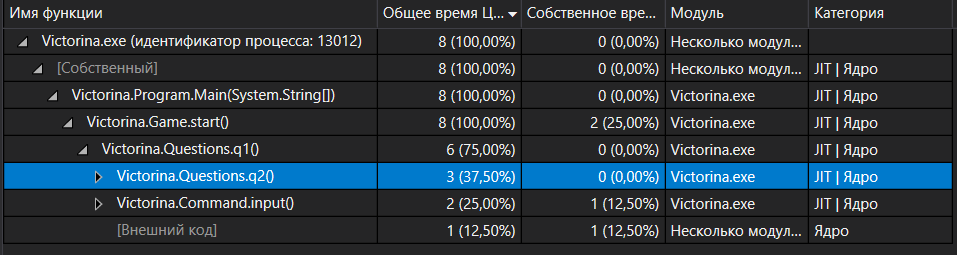


В этом представлении выбранная функции отображается в заголовке и в поле **Текущая функция** (в этом примере q1). Функция, вызывавшая текущую функцию, отображается в левой части окна в разделе **Вызывающие функции**, а все функции, вызываемые текущей функцией, отображаются в поле **Вызываемые функции** справа. (Можно выбрать любое поле, чтобы изменить текущую функцию.)

В этом представлении показано общее время (мс) и доля общего времени выполнения приложения, затраченного на выполнение функции. В поле **Тело функции** также показан общий объем времени (и доля времени), затраченного в теле функции за исключением времени, затраченного в вызываемых и вызывающих функциях.

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

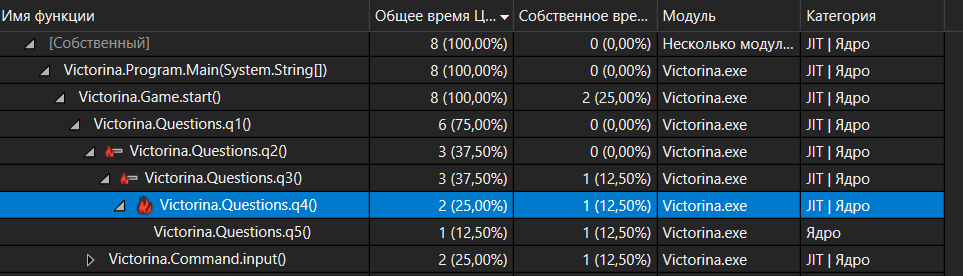
Каждая нумерованная область на рисунке соответствует определенному шагу в процедуре.



Ниже приведены дополнительные сведения по значениям столбца.

* + Значение **Общее время ЦП** указывает, какой объем работы был выполнен некоторой функцией и всеми вызванными ей функциями. Высокие значения общего времени ЦП указывают на функции, которые в целом являются наиболее ресурсоемкими.
  + Значение **Собственное время** указывает, какой объем работы был выполнен кодом в теле функции, за исключением работы, выполненной вызванными ей функциями. Высокие значения **собственного времени ЦП** могут свидетельствовать о проблемах производительности внутри самой функции.
  + **Модули**. Имя модуля, содержащего функцию, или количество модулей, содержащих функции в узле [Внешний код].

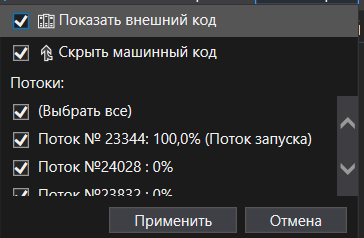
Чтобы увидеть вызовы функций, которые используют самый высокий процент ЦП в представлении дерева вызовов, нажмите **Развернуть критический путь**.



### Просмотр внешнего кода

Внешний код — это функции в компонентах системы и платформы, которые исполняются вашим кодом. Внешний код включает функции, которые запускают и останавливают приложение, отрисовывают пользовательский интерфейс, управляют потоками и предоставляют приложению другие низкоуровневые службы. В большинстве случаев внешний код вас интересовать не будет, поэтому средство "Загрузка ЦП" собирает внешние функции пользовательского метода в один узел **[Внешний код]** .

Если вы захотите посмотреть пути к вызовам внешнего кода, выберите **Показать внешний код** в списке **Представление фильтра** и выберите **Применить**.



Помните о том, что многие цепочки вызовов имеют глубокий уровень вложенности, поэтому ширина столбца "Имя функции" может превышать ширину многих мониторов. В этом случае имена функций отображаются в виде **[…]** .

Используйте поле поиска, чтобы найти требуемый узел, а затем воспользуйтесь горизонтальной полосой прокрутки для отображения данных в представлении.