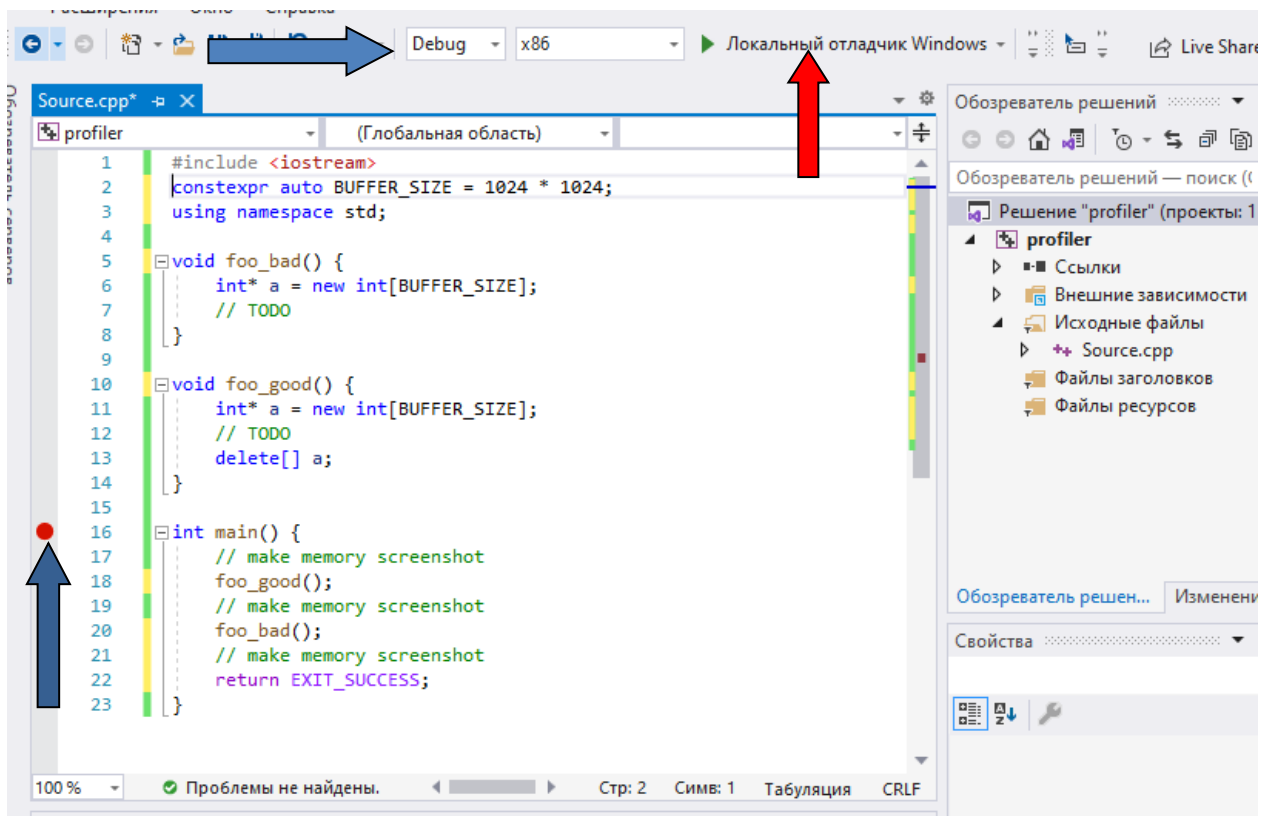


Профилирование памяти в IDE MS VisualStudio 2019

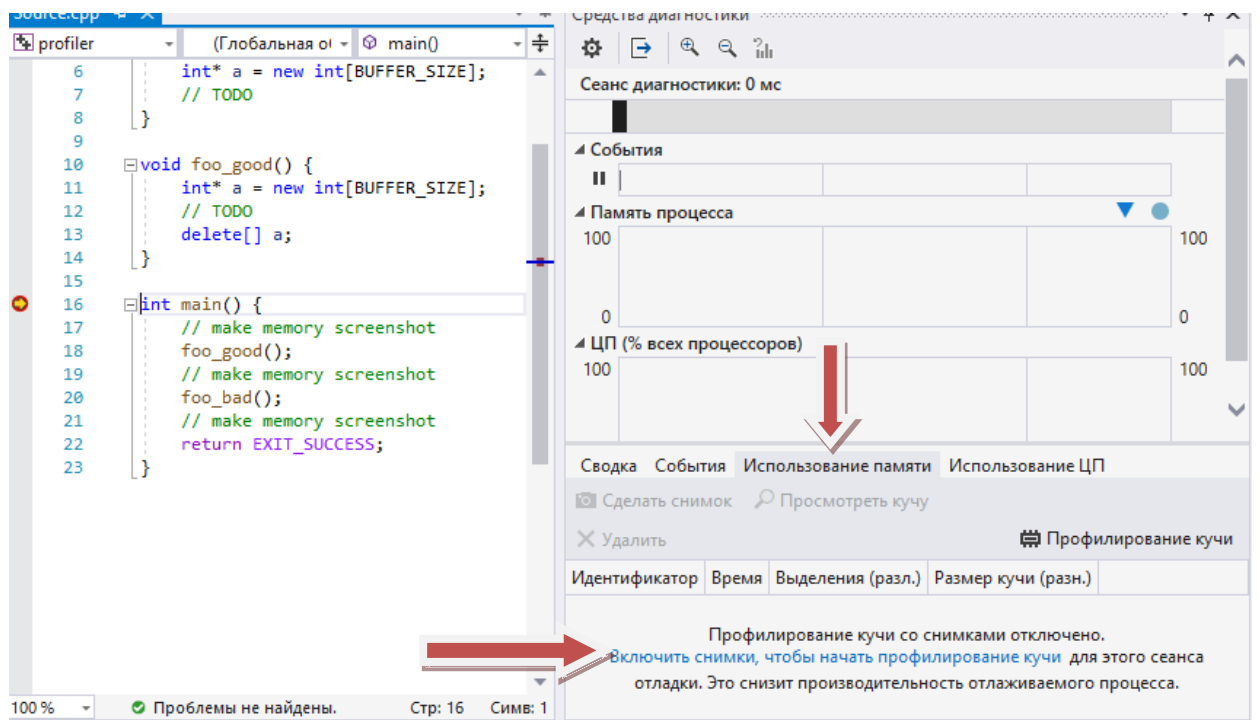
При отладке программ очень часто возникает необходимость в профилировании выделяемой памяти в программе. Иногда для этого используют специальные библиотеки для профилирования, такие как CRT (почитать можно [ТУТ](#)). Если не нужна точная оценка выделяемой памяти, можно воспользоваться и функционалом стандартного средства отладки программы в MS VS.

Для примера напишем такую программу:



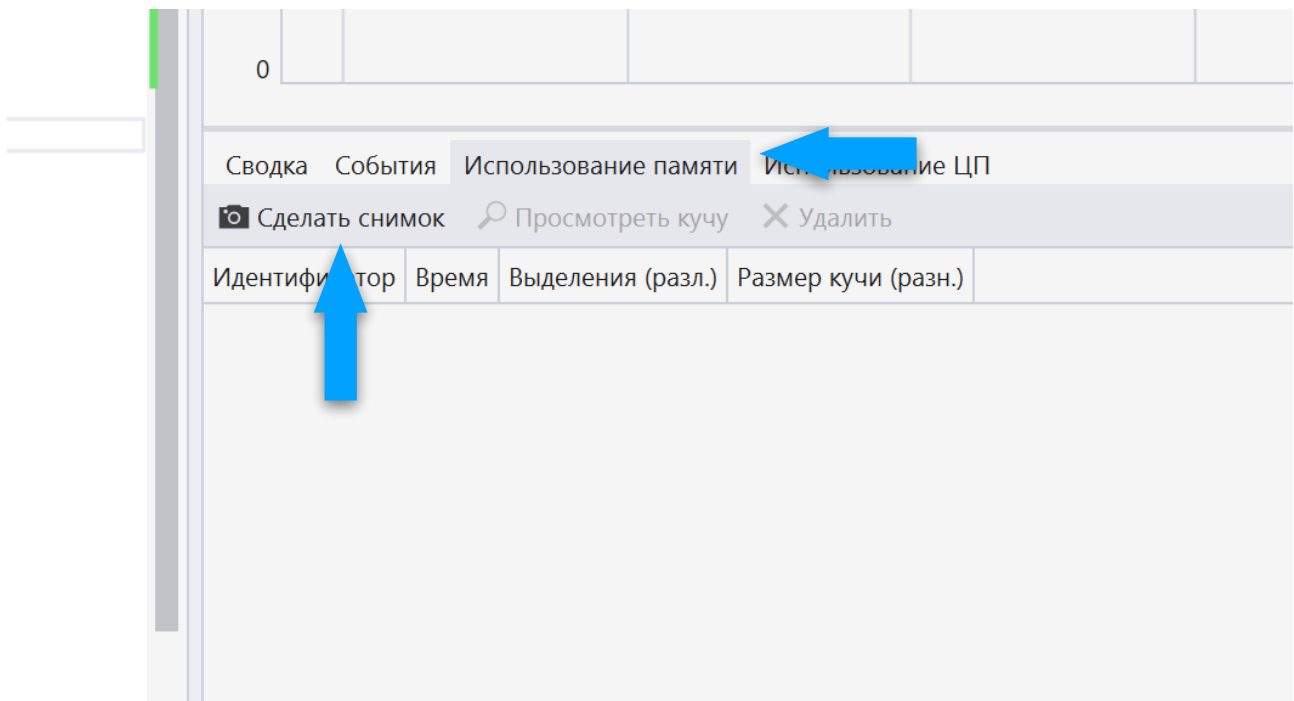
В данном примере в функции `foo_good()` память выделяется и высвобождается, а в функции `foo_bad()` специально сделана ошибка, память не высвобождается, что приводит к утечке памяти.

Для профилирования поставим метку отладки на функции `main()` (16 строка) и выберем сборку проекта в режиме отладки (Debug). Далее запустим локальный отладчик (красная стрелка на предыдущем скриншоте) и в окне справа откроется меню профилирования. Если окно не открылось, вызовите его через пункт меню «Отладка - Окна - Показать средства диагностики» или комбинацией клавиш `Ctrl+Alt+F2`:



Найдите вкладку «использование памяти» И включите режим получения снимков памяти для профилирования кучи (при повторных запусках включать снимки не требуется).

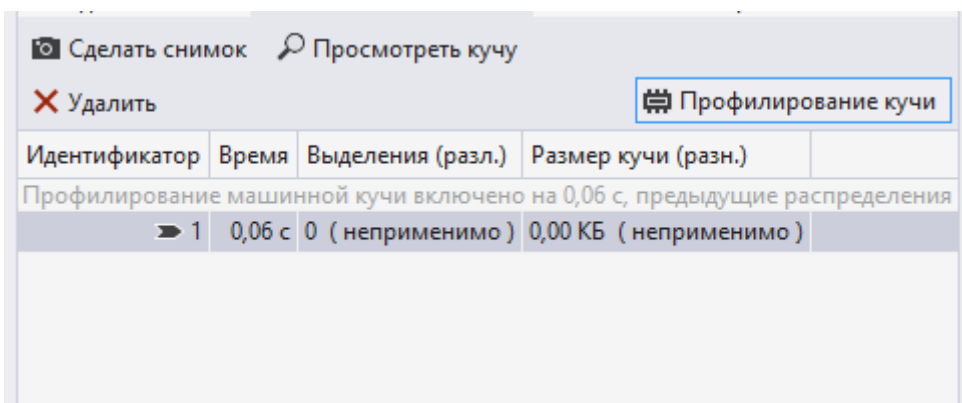
В окне отладки у Вас в разделе «Использование памяти» появится кнопка «Сделать снимок»:



Под этой кнопкой находится таблица, которая сначала будет пустой и в

ней будет появляться новая строка, каждый раз, когда вы будете нажимать кнопку «Сделать снимок». Каждая запись отображает состояние памяти в данный момент программы.

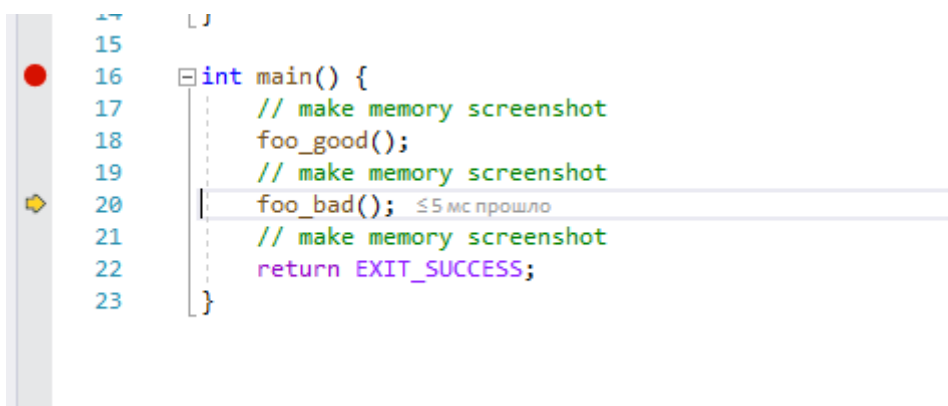
Для отладки утечек сделаем снимок памяти в начале программы (данные о начальном размере кучи могут отличаться от приведённых на снимке, особенно при повторных запусках:



The screenshot shows the 'Профилирование кучи' (Heap Profiling) window in Visual Studio. It contains a table with the following data:

Идентификатор	Время	Выделения (разл.)	Размер кучи (разн.)
Профилирование машинной кучи включено на 0,06 с, предыдущие распределения			
1	0,06 с	0 (неприменимо)	0,00 КБ (неприменимо)

Появилась информация о начальном состоянии памяти. Далее пройдем отладчиком до вызова функции `foo_good` и также выполним её (шаг с заходом F11):



```
15
16 int main() {
17     // make memory screenshot
18     foo_good();
19     // make memory screenshot
20     foo_bad(); ≤5 мс прошло
21     // make memory screenshot
22     return EXIT_SUCCESS;
23 }
```

После прохода функции сделаем снимок программы. Внутри данной функции выделялось и высвобождалось 4 мегабайта памяти, но можно видеть, что утечки памяти нет:

Идентификатор	Время	Выделения (разл.)	Размер кучи (разн.)
1	0,03 с	0 (неприменимо)	0,00 КБ (неприменимо)
2	0,04 с	0 (+0)	0,00 КБ (+0,00 КБ)

Теперь сделаем то же самое после функции `foo_bad()` и, дойдя до конца программы, тоже сделаем снимок памяти (синяя стрелка на скриншоте снизу):

```

15
16 int main() {
17     // make memory screenshot
18     foo_good();
19     // make memory screenshot
20     foo_bad();
21     // make memory screenshot
22     return EXIT_SUCCESS;
23 }

```

Идентификатор	Время	Выделения (разл.)	Размер кучи (разн.)
1	0,00 с	0 (неприменимо)	0,00 КБ (неприменимо)
2	0,01 с	0 (+0)	0,00 КБ (+0,00 КБ)
3	0,01 с	1 (+1 ↑)	4 096,04 КБ (+4 096,04 КБ ↑)
4	0,01 с	1 (+0)	4 096,04 КБ (+0,00 КБ)

Красная стрелка на скриншоте сверху указывает на состояние памяти после вызова функции `foo_bad()`, а зелёная, на состояние при завершении программы. Можно сделать вывод, что при выполнении программы выделяется лишние 4 мб памяти, причём в функции `foo_bad()`.

Если нужно узнать, где именно в программе произошло выделение данной памяти. Для этого необходимо нажать левой кнопкой мыши на размер памяти (синяя стрелка на скриншоте выше):

После чего откроется окно с состоянием кучи в выбранный момент (моментальный снимок):

Тип объекта	Счетчик	Размер (байт)
int[]	1	4 194 304

Идентификатор	Время	Выделения (разл.)	Размер кучи (разн.)
1	0,01 с	160 (неприменимо)	49,30 КБ (неприменимо)
2	0,01 с	160 (+0)	49,30 КБ (+0,00 КБ)
3	0,01 с	161 (+1 ↑)	4 145,34 КБ (+4 096,04 КБ ↑)
4	0,01 с	161 (+0)	4 145,34 КБ (+0,00 КБ)

Можно заметить, что создана запись об указателе на выделенную память. Далее можно перейти к самому экземпляру, для этого выберите нужную строку и нажмите левой кнопкой мыши на значок, как указано стрелкой на скриншоте ниже:

Режим просмотра: Представление типов

Тип объекта	Счетчик	Размер (байт)
int[]	1	4 194 304

↑

Теперь, нажав на нужный экземпляр, на стеке вызовов можно найти место в программе, где была выделена память (синяя стрелка на скриншоте ниже):

← Экземпляры int[]

Моментальный снимок не отражает текущее состояние программы. Значения недоступны.

Экземпляр	Размер (байт)	Возраст (мс)
<0xCF0040>	4 194 304	3056,907000

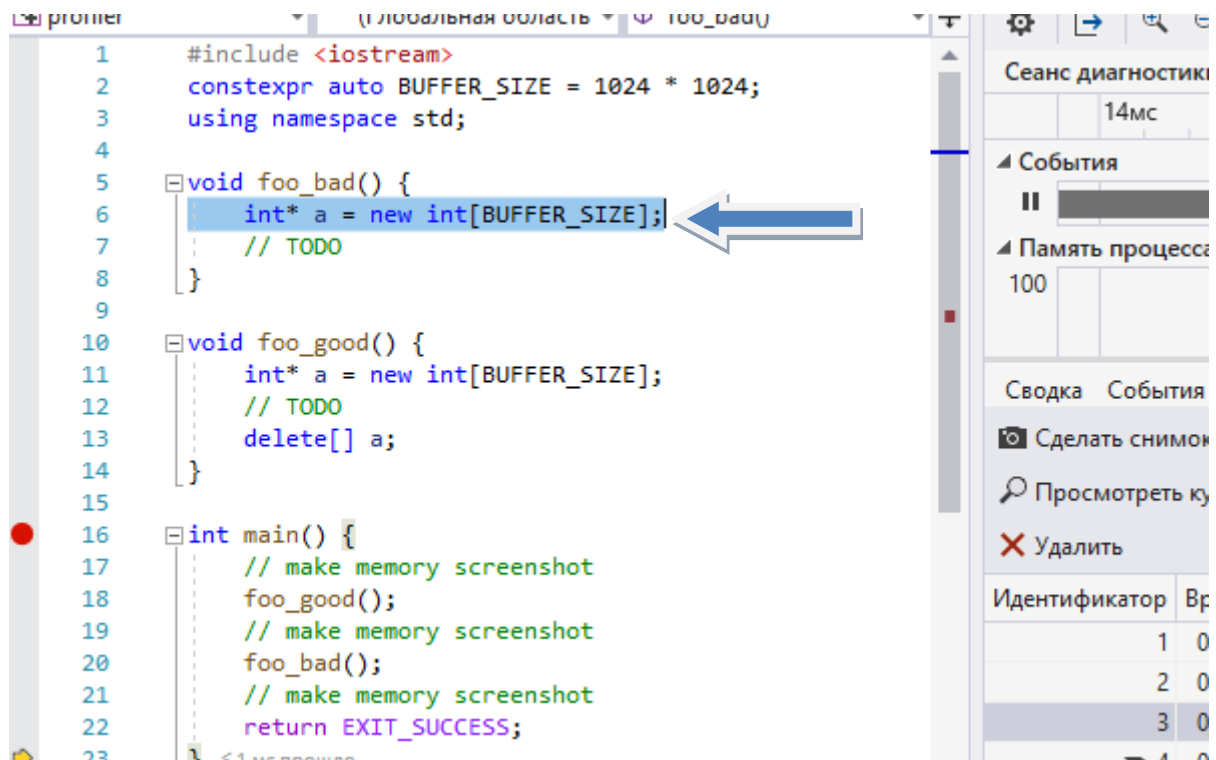
↑

Стек вызовов выделений

Имя
profiler.exe!foo_bad() — строка 6
profiler.exe!main() — строка 22
profiler.exe!invoke_main() — строка 78

← Фреймы стека

Дважды нажав на строку в стеке вызовов MS VS откроет окно с кодом программы и выделит оператор выделения данной памяти (синяя стрелка на скриншоте ниже):



Для более детального профилирования необходимо делать шаг с заходом в функцию и после каждой инструкции вызывать снимок состояния кучи.