**Отчёт по практике 4.**

**Студент: Гуртякова А.П.**

**Задача:**Изучить сборщик мусора

**Цель задания:**Описать понятие и работу сборщика мусора

**Ответить на вопросы:**

Что такое Сборщик мусора?

Опишите его работу.

**Ответ:**

В среде CLR сборщик мусора выполняет функции автоматического диспетчера памяти. Сборщик мусора управляет выделением и освобождением памяти для приложения. Поэтому разработчикам, работающим с управляемым кодом, не нужно писать код для выполнения задач управления памятью. Автоматическое управление памятью может устранить распространенные проблемы, такие как забыть освободить объект и вызвать утечку памяти или попытку доступа к освобожденной памяти для объекта, который уже был освобожден.

Механизм оптимизации сборщика мусора определяет наилучшее время для выполнения сбора, основываясь на произведенных выделениях памяти. Когда сборщик мусора выполняет очистку, он освобождает память, выделенную для объектов, которые больше не используются приложением. Он определяет, какие объекты больше не используются, анализируя *корни* приложения. Корни приложения содержат статические поля, локальные переменные в стеке потока, регистры процессора, дескрипторы сборки мусора и очередь завершения. Каждый корень либо ссылается на объект, находящийся в управляемой куче, либо имеет значение NULL. Сборщик мусора может запросить остальную часть среды выполнения для этих корней. Сборщик мусора использует этот список для создания графа, содержащего все объекты, доступные из корней.

Объекты, которых нет в графе, недоступны из корней приложения. Сборщик мусора считает недостижимые объекты мусором и освобождает выделенную для них память. В процессе очистки сборщик мусора проверяет управляемую кучу, отыскивая блоки адресного пространства, занятые недостижимыми объектами. При обнаружении недостижимого объекта он использует функцию копирования памяти для уплотнения достижимых объектов в памяти, освобождая блоки адресного пространства, выделенные под недостижимые объекты. После уплотнения памяти, занимаемой достижимыми объектами, сборщик мусора вносит необходимые поправки в указатель, чтобы корни приложения указывали на новые расположения объектов. Он также устанавливает указатель управляемой кучи в положение после последнего достижимого объекта.

Память уплотняется, только если при очистке обнаруживается значительное число недостижимых объектов. Если все объекты в управляемой куче сохраняют коллекцию, сжатие памяти не требуется.

Для повышения производительности среда выполнения выделяет память для больших объектов в отдельной куче. Сборщик мусора автоматически освобождает память, выделенную для больших объектов. Но для устранения перемещений в памяти больших объектов эта память обычно не сжимается.

Сборка мусора возникает при выполнении одного из следующих условий:

* Недостаточно физической памяти в системе. Размер памяти определяется уведомлением о нехватке памяти от операционной системы или нехватке памяти, как указано узлом.
* Объем памяти, используемой объектами, выделенными в управляемой куче, превышает допустимый порог. Этот порог непрерывно корректируется во время выполнения процесса.
* Вызывается метод GC.Collect . Почти во всех случаях не нужно вызывать этот метод, так как сборщик мусора работает непрерывно. Этот метод в основном используется для уникальных ситуаций и тестирования.

Объекты, которые не были восстановлены в сборке мусора, называются выжившими и повышаются до следующего поколения:

* Объекты, оставшиеся после сборки мусора поколения 0, подвигаются в поколение 1.
* Объекты, оставшиеся после сборки мусора поколения 1, подвигаются в поколение 2.
* Объекты, оставшиеся после сборки мусора поколения 2, остаются в поколении 2.

Когда сборщик мусора обнаруживает высокую долю выживания в поколении, он повышает порог распределений для этого поколения. При следующей сборке мусора освобождается заметная часть занятой памяти. В среде CLR непрерывно контролируется равновесие двух приоритетов: не позволить рабочему набору приложения стать слишком большим, задерживая сборку мусора, и не позволить сборке мусора выполняться слишком часто.