Распределение памяти

Распределение памяти - это процесс, в результате которого отдельным элементам исходной программы ставятся в соответствие адрес, размер и атрибуты области памяти, необходимой для размещения лексических единиц.

Область памяти - это блок ячеек памяти, выделяемых для данных и каким-то образом объединенных логически.

Распределение памяти выполняется после фазы анализа текста исходной программы **на этапе подготовки к генерации объектного модуля** (перед генерацией кода объектного модуля).

Исходными данными для процесса распределения памяти служат сведения о семантике конструкций ЯП, таблица идентификаторов, построенная лексическим анализатором и информация, полученная синтаксическим анализатором при анализе декларативной чисти программы.

Современные компиляторы, в основном, работают с относительными, а не с абсолютными адресами ячеек памяти.

Распределение памяти

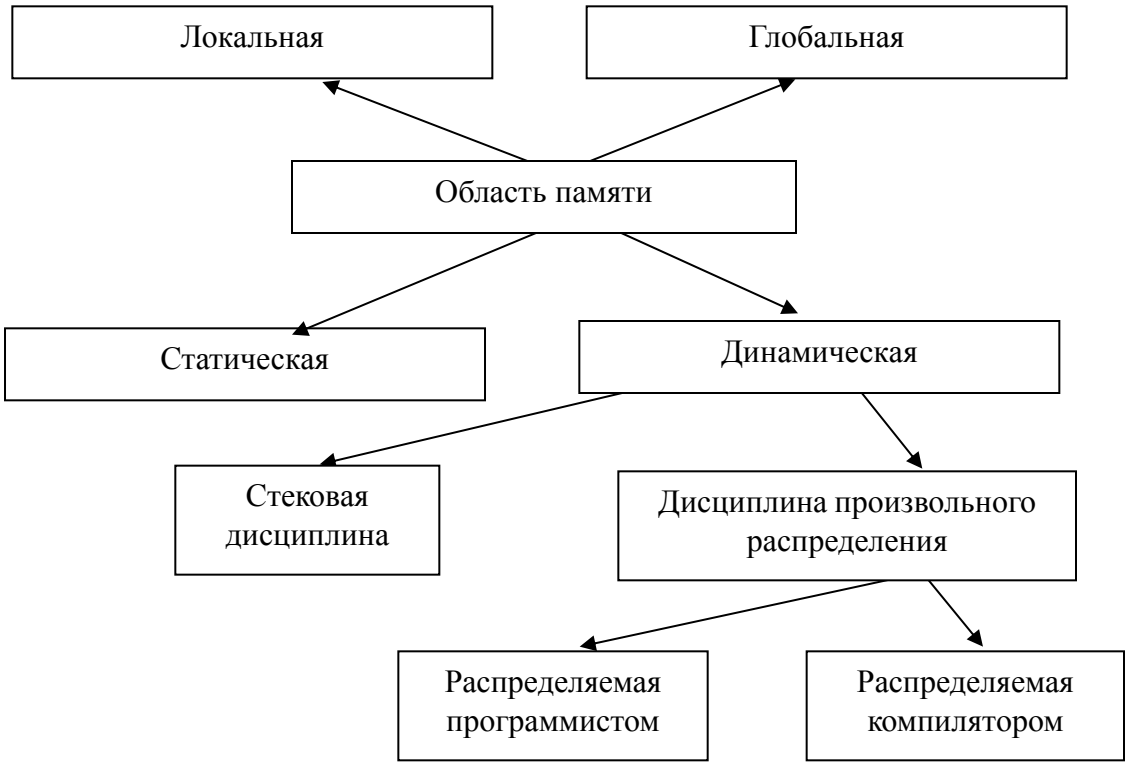
Семантика программ подразумевает, что при их выполнении области памяти будут необходимы для хранения:

- кодов пользовательских программ;
- данных, необходимых для работы этих программ;
- кодов системных программ, обеспечивающих поддержку пользовательских программ в период их выполнения;
- записей о текущем состоянии процесса выполнения программ (например, записей об активации процедур).

По **способу использования** области памяти делятся на ***глобальные*** и ***локальные***, а по **способу распределения** – на ***статические*** и ***динамические***.

Классы памяти

Выделяемую память можно разделить на локальную / глобальную и статическую / динамическую.



Классы памяти

Локальная память - это область памяти, которая выделяется в начале выполнения некоторого фрагмента результирующей программы (блока, функции, оператора…) и может быть освобождена по завершении выполнения данного фрагмента. Доступ к локальной области памяти всегда запрещен за пределами того фрагмента программы, в котором она выделяется.

Глобальная память - это область памяти, которая выделяется один раз при инициализации результирующей программы и действует все время выполнения программы. Как правило, глобальная область памяти доступна из любой части исходной программы.

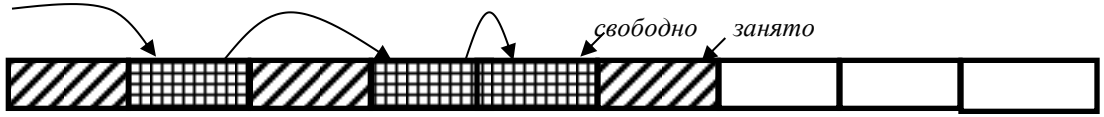
Статическая память - это область памяти, размер которой известен на этапе компиляции. Для статической памяти компилятор порождает некоторый адрес (как правило, относительный), и дальнейшая работа с ней происходит относительно этого адреса.

Динамическая память - это область памяти, размер которой становится известным только на этапе выполнения результирующей программы. Для динамической памяти компилятор порождает фрагмент кода, который отвечает за распределение памяти (ее выделение и освобождение). Как правило, с динамическими областям памяти связаны многие операции с указателями и с экземплярами объектов (классов) в ООЯП.

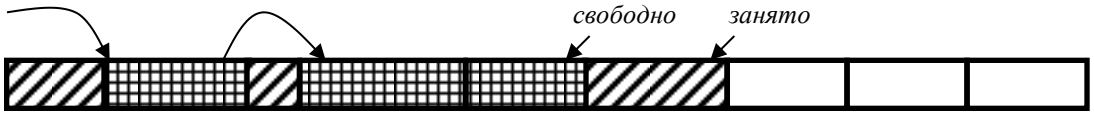
Динамические области памяти можно разделить на динамические области памяти, **выделяемые пользователем** и **непосредственно компилятором**.

Методы динамического распределения памяти

Явное выделение блоков фиксированного размера:



Явное выделение блоков переменного размера:



Неявное выделение динамической памяти

Неявно выделяемые блоки памяти могут быть фиксированного или переменного размера.

При неявном динамическом выделении и освобождении блоков памяти, выделяемые блоки обычно имеют следующую структуру:

- размер блока (если блок фиксированного размера, то эта информация в нём не хранится.);
- счётчик ссылок, пометка (обычно есть либо одно, либо другое):

Счетчик ссылок подсчитывает количество указателей в программе, которые ссылаются на этот блок (например, при $p = q$, один счётчик (для блока, на который указывал указатель p) уменьшается на единицу, а другой увеличивается), если счётчик равен нулю, то блок не используется и его можно освободить. Существует проблема циклических ссылок, когда счётчики всегда > 0.

Пометка фиксирует, задействован данный блок или не задействован, т.е. у программы есть хотя бы один указатель, ссылающийся на этот блок. В некоторый момент начинает работать «сборщик мусора». Он помечает все блоки как недостижимые, а затем начинает анализ текущих указателей программы, что является очень трудоемким процессом. Блоки, на которые ничего не указывает, считаются свободными.

- указатели на блоки
- заказанная память, необходимая для размещения объектов программы.

Общие принципы генерации объектного кода

При генерации объектного кода компилятор переводит текст программы во внутреннем представлении в текст программы на выходном языке (как правило, машинном).

Генерация объектного кода происходит на основе:

- определенной на фазе анализа компиляции синтаксической структуры программы,
- информации, хранящейся в таблице идентификаторов,
- результата распределения памяти.

Характер и сложность отображения промежуточного представления программы в последовательность команд на машинном языке зависит от языка внутреннего представления и архитектуры вычислительной системы, на которую ориентирована результирующая программа.

Часто для построения кода результирующей программы компиляторы используют синтаксически управляемый перевод.