**Исходный, Машинный, Объектный коды и Исполняемые файлы:**

**Исходный код** или исходный текст - текст на языке программирования, который пишет человек (как правило, человек). Это обычный текстовый файл (как правило).  
  
Но процессоры в компьютера/смартфонах и пр. - довольно тупые устройства, способные понимать только чрезвычайно простой машинный код.  
Поэтому любая программа прямо или косвенно превращается из исходного кода в машинный и только после этого исполняется компьютером/смартфоном и пр.  
  
**Объектный код** - это промежуточное представление кода. Еще не машинный, но уже не исходный. Используется на этапе сборки программы из нескольких кусков (возможно, написанных в исходном коде разными людьми и в разное время).  
  
**Машинный код** - уже готов для исполнения. То есть просто берете его, ничего не нужно более. Делаете двойной щелчок мыши, к примеру, и программа запускается

**Исполняемыми файлами** называются файлы, содержащие в себе готовые к запуску компьютерные программы. В зависимости от того, в каком виде записана программа в исполняемом файле, выделяют две большие подгруппы файлов данного класса. Первая подгруппа - это бинарные файлы; они содержат в себе запись программы в виде специальных машинных кодов, которые, чаще всего, передаются непосредственно процессору. Такие файлы ещё сокращённо называют бинарниками. В противоположность им существуют исполняемые файлы, в которых программа записана в виде текста - то есть, в том самом виде, в каком с ней работал создавший её программист. Такие исполняемые файлы называют скриптами. Также этот термин нередко употребляют и по отношению к самим программам, которые хранятся в подобных файлах.  Самый часто встречающийся их вид - это приложение. Приложения имеют расширения EXE и могут запускаться самостоятельно. Помимо них, существуют динамические библиотеки (их расширение - DLL), которые содержат в себе общие для разных приложений функции. Ещё есть драйверы (DRV или VXD) - специальные программы, нужные для того, чтобы система могла взаимодействовать с конкретными моделями тех или иных устройств. Исполняемые файлы (в Windows особенно) могут зависеть друг от друга: например, для запуска любого приложения нужны определённые системные динамические библиотеки, а им, в свою очередь, нужны драйверы.

Следует заметить, что исполняемые файлы содержат в себе не только непосредственно сами программы, но и разные дополнительные данные. Это могут быть различные графические ресурсы, отображаемые программой, тексты надписей, описания диалоговых окон и т.п. Ярким примером этого могут служить самораспаковывающиеся архивы, которые содержат внутри себя большие объёмы упакованной с целью уменьшения её объёма при передаче или хранении информации.

# Транслятор, компилятор, интерпретатор

***Транслятор***(англ. translator — переводчик) — это программа-переводчик. Она преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд. Транслятор обычно выполняет также диагностику ошибок, формирует словари идентификаторов, выдаёт для печати тексты программы и т. д. Язык, на котором представлена входная программа, называется исходным языком, а сама программа — исходным кодом. Выходной язык называется целевым языком или объектным кодом.

Трансляторы реализуются в виде компиляторов или интерпретаторов. С точки зрения выполнения работы компилятор и интерпретатор существенно различаются.

***Компилятор***(англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется. Входной информацией для компилятора (исходный код) является описание алгоритма или программа на проблемно-ориентированном языке, а на выходе компилятора — эквивалентное описание алгоритма на машинно-ориентированном языке (объектный код).

***Интерпретатор***(англ. interpreter — истолкователь, устный переводчик) переводит и выполняет программу строка за строкой, производит пооператорный (покомандный, построчный) анализ, обрабатывает и тут же выполняет исходную программу или запроса (в отличие от компиляции, при которой программа транслируется без её выполнения.

+После того, как программа откомпилирована, ни сама исходная программа, ни компилятор более не нужны. В то же время программа, обрабатываемая интерпретатором, должна заново переводиться на машинный язык при каждом очередном запуске программы.

Откомпилированные программы работают быстрее, но интерпретируемые проще исправлять и изменять.

Разница в том, что компилятор преобразует код алгоритмического языка в исполняемый (байт-код либо нативный) код (в пределе - в машинные коды), в то время как интерпретатор самостоятельно распознаёт и выполняет построчно инструкции программного кода.

Таким образом, можно заключить, что компилятор приобразует исходный код к близкому к машинному коду. Интерпретатор же, позволяет выполнять некоторое представление программы, налету преобразуя в машинный код.

Препроцессор. Директивы препроцессора. Примеры

**Препроцессор С/С++** — программа, подготавливающая код программы на языке [C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F))/[C++](https://ru.wikipedia.org/wiki/C%2B%2B) к [компиляции](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80).

[Препроцессором](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) выполняются следующие действия:

* удаление экранированных [символов перевода строки](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B2%D0%BE%D0%B4_%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8);
* замена строчных и блочных [комментариев](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%80%D0%B8%D0%B8_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) пустыми строками (с удалением окружающих пробелов и символов табуляции);
* вставка (включение) содержимого произвольного файла (#include);
* [макроподстановки](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D1%80%D0%BE%D1%81) (#define);
* условная [компиляция](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) (#if, #ifdef, #elif, #else, #endif);
* вывод сообщений (#warning, #error).

**Директивой** (**командной строкой**) препроцессора называется строка в исходном коде, имеющая следующий формат: #ключевое\_слово параметры:

* символ [#](https://ru.wikipedia.org/wiki/%D0%9E%D0%BA%D1%82%D0%BE%D1%82%D0%BE%D1%80%D0%BF);
* ноль или более символов пробелов и/или табуляции;
* одно из предопределённых ключевых слов;
* параметры, зависимые от ключевого слова.

Список ключевых слов:

* define — создание константы или макроса;
* undef — удаление константы или макроса;
* include — вставка содержимого указанного файла;
* if — проверка истинности выражения;
* ifdef — проверка существования константы или макроса;
* ifndef — проверка несуществования константы или макроса;
* else — ветка условной компиляции при ложности выражения if;
* elif — проверка истинности другого выражения; краткая форма записи для комбинации else и if;
* endif — конец ветки условной компиляции;
* line — указание имени файла и номера текущей строки для компилятора;
* error — вывод сообщения и остановка компиляции;
* warning — вывод сообщения без остановки компиляции;
* pragma — указание действия, зависящего от реализации, для препроцессора или компилятора;
* если ключевое слово не указано, директива игнорируется;
* если указано несуществующее ключевое слово, выводится сообщение об ошибке и компиляция прерывается. (В некоторых компиляторах, таких как g++, компиляция продолжается, просто показывая предупреждение.)

Парадигмы программирования

**Паради́гма программи́рования** — это совокупность идей и понятий, определяющих стиль написания [компьютерных программ](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)