**Ассемблер**

Машинно-ориентированный [язык низкого уровня](http://ru.wikipedia.org/wiki/%D0%9D%D0%B8%D0%B7%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%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) с командами, обычно соответствующими [командам машины](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8), который может обеспечить дополнительные возможности вроде [макрокоманд](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D1%80%D0%BE%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%B0); автокод, расширенный конструкциями[языков программирования высокого уровня](http://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%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), такими как выражения, макрокоманды, средства обеспечения [модульности программ](http://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C_(%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)).  
Автокод — язык программирования, предложения которого по своей структуре в основном подобны командам и обрабатываемым данным конкретного [машинного языка](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4).

Достоинства:

* Язык ассемблера позволяет писать самый быстрый и компактный код, какой вообще возможен для данного процессора.
* Если код программы достаточно большой, — данные, которыми он оперирует, не помещаются целиком в [регистрах процессора](http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80%D0%B0), то есть частично или полностью находятся в [оперативной памяти](http://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C), — то искусный программист, как правило, способен значительно оптимизировать программу по сравнению с транслятором с языка высокого уровня по одному или нескольким параметрам и создать код близкий к [оптимальному по Парето](http://ru.wikipedia.org/wiki/%D0%AD%D1%84%D1%84%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BF%D0%BE_%D0%9F%D0%B0%D1%80%D0%B5%D1%82%D0%BE) (как правило, быстродействие программы достигается за счет удлинения кода и наоборот):
  + скорость работы — за счёт оптимизации вычислительного алгоритма и/или более рационального обращения к [оперативной памяти (ОП)](http://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) (например, если все исходные данные хранятся в регистрах процессора, то можно исключить излишние обращения к ОП), перераспределения данных, табличного вычисления функций;
  + объём кода (в том числе за счёт эффективного использования промежуточных результатов). Сокращение объёма кода также нередко повышает скорость выполнения программы.
* Обеспечение максимального использования специфических возможностей конкретной платформы, что также позволяет создавать более эффективные программы, в том числе менее ресурсоёмкие.
* При программировании на языке ассемблера возможен непосредственный доступ к аппаратуре, и, в частности, [портам ввода-вывода](http://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D1%82_%D0%B2%D0%B2%D0%BE%D0%B4%D0%B0-%D0%B2%D1%8B%D0%B2%D0%BE%D0%B4%D0%B0), регистрам процессора и др. Во многих операционных системах прямое обращение из прикладных программ для записи в регистры периферийного оборудования блокировано для надёжности работы системы и исключения "зависаний".
* Язык ассемблера часто применяется для создания [драйверов](http://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80) оборудования и [ядра операционной системы](http://ru.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) (или машиннозависимых подсистем ядра ОС), тогда, когда важно временно́е согласование работы периферийных устройств с центральным процессором.
* Язык ассемблера используется для создания «[прошивок](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%88%D0%B8%D0%B2%D0%BA%D0%B0)» [BIOS](http://ru.wikipedia.org/wiki/BIOS).
* С помощью языка ассемблера часто создаются машиннозависимые подпрограммы [компиляторов](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) и [интерпретаторы](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80) [языков высокого уровня](http://ru.wikipedia.org/wiki/%D0%AF%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_%D0%B2%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D0%B3%D0%BE_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D1%8F), а также реализуется совместимость[платформ](http://ru.wikipedia.org/wiki/%D0%90%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%B0).
* С помощью программы [дизассемблера](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B7%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80) можно понять алгоритмы работы исследуемой программы при отсутствии листинга на высокоуровневом языке, изучая только машинные коды, но, в сложных нетривиальных программах, это очень и очень трудоёмко.

Недостатки:

* В силу машинной ориентации («низкого» уровня) языка ассемблера человеку сложнее читать и понимать программу на нём по сравнению с языками программирования высокого уровня; программа состоит из слишком «мелких» элементов — машинных команд, соответственно, усложняются программирование и отладка, растут трудоёмкость и вероятность внесения ошибок.
* Требуется повышенная квалификация программиста для получения качественного кода: код, написанный средним программистом на языке ассемблера, обыкновенно оказывается не лучше или даже хуже кода, порождаемого [оптимизирующим компилятором](http://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D0%BC%D0%B8%D0%B7%D0%B8%D1%80%D1%83%D1%8E%D1%89%D0%B8%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) для сравнимых программ, написанных на языке высокого уровня[[5]](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0#cite_note-5).
* Программа на языке высокого уровня может быть перекомпилирована с автоматической оптимизацией под особенности новой целевой платформы[[6]](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0#cite_note-6), программа же на языке ассемблера на новой платформе может потерять своё преимущество в скорости без ручного переписывания кода[[7]](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0#cite_note-7)[[8]](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0#cite_note-8).
* Как правило, меньшее количество доступных [библиотек](http://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%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)) по сравнению с современными индустриальными языками программирования.
* Отсутствует переносимость программ на компьютеры с другой архитектурой и системой команд.

На языке ассемблера пишут программы или их фрагменты в тех случаях, когда критически важны:

* быстродействие ([драйверы](http://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80), игры);
* объём используемой памяти (загрузочные секторы, встраиваемое ([англ.](http://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *embedded*) программное обеспечение, программы для [микроконтроллеров](http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%BB%D0%B5%D1%80) и процессоров с ограниченными ресурсами, [вирусы](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B8%D1%80%D1%83%D1%81), программные защиты).

### Языки программирования низкого уровня

Первые компьютеры приходилось программировать двоичными машинными кодами. Однако программировать таким образом - довольно трудоемкая и тяжелая задача. Для упрощения этой задачи начали появляться языки программирования низкого уровня, которые позволяли задавать машинные команды в понятном для человека виде. Для преобразования их в двоичный код были созданы специальные программы - трансляторы.

Трансляторы делятся на:

[компиляторы](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D1%8B) - превращают текст программы в машинный код, который можно сохранить и после этого использовать уже без компилятора (примером является исполняемые файлы с расширением \*.exe) .

[интерпретаторы](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80) - превращают часть программы в машинный код, выполняют его и после этого переходят к следующей части. При этом каждый раз при выполнении программы используется интерпретатор .

Примером языка низкого уровня является [ассемблер](http://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80_(%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)). Языки низкого уровня ориентированы на конкретный тип [процессора](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) и учитывают его особенности, поэтому для переноса программы на ассемблере на другую аппаратную платформу ее нужно почти полностью переписать. Определенные различия есть и в синтаксисе программ под разные компиляторы. Правда, [центральные процессоры](http://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) для компьютеров фирм [AMD](http://ru.wikipedia.org/wiki/AMD) и [Intel](http://ru.wikipedia.org/wiki/Intel" \o "Intel) практически совместимы и отличаются лишь некоторыми специфическими командами. А вот специализированные процессоры для других устройств, например, видеокарт и телефонов содержат существенные различия.

Языки низкого уровня, как правило, используют для написания небольших системных программ, драйверов устройств, модулей стыков с нестандартным оборудованием, программирование специализированных микропроцессоров, когда важнейшими требованиями являются компактность, быстродействие и возможность прямого доступа к аппаратным ресурсам.

[Ассемблер](http://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80_(%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)) - язык низкого уровня, широко применяется до сих пор.

### Языки программирования высокого уровня

Особенности конкретных компьютерных архитектур в них не учитываются, поэтому созданные приложения легко переносятся с компьютера на компьютер. В большинстве случаев достаточно просто перекомпилировать программу под определенную компьютерную архитектурную и операционную систему. Разрабатывать программы на таких языках значительно проще и ошибок допускается меньше. Значительно сокращается время разработки программы, что особенно важно при работе над большими программными проектами .

Сейчас в среде разработчиков считается, что языки программирования, которые имеют прямой доступ к памяти и регистров или имеют ассемблерные вставки, нужно считать языками программирования с низким уровнем абстракции. Поэтому большинство языков, считавшихся языками высокого уровня до 2000 года сейчас уже таковыми не считаются.

* [Адресный язык программирования](http://ru.wikipedia.org/wiki/%D0%90%D0%B4%D1%80%D0%B5%D1%81%D0%BD%D1%8B%D0%B9_%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)
* [Фортран](http://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D1%82%D1%80%D0%B0%D0%BD)
* [Кобол](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B1%D0%BE%D0%BB)
* [Алгол](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D0%BB)
* [Pascal](http://ru.wikipedia.org/wiki/Pascal)
* [Java](http://ru.wikipedia.org/wiki/Java)
* [C](http://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++](http://ru.wikipedia.org/wiki/C%2B%2B)
* [Objective C](http://ru.wikipedia.org/wiki/Objective-C)
* [Smalltalk](http://ru.wikipedia.org/wiki/Smalltalk)
* [C#](http://ru.wikipedia.org/wiki/C_Sharp)
* [Delphi](http://ru.wikipedia.org/wiki/Delphi_(%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))

Недостатком языков высокого уровня является большой размер программ по сравнению с программами на языках низкого уровня. Сам текст программ на языке высокого уровня меньше, однако, если взять в байтах, то код, изначально написанный на ассемблере, будет более компактным. Поэтому в основном языки высокого уровня используются для разработки программного обеспечения компьютеров и устройств, которые имеют большой объем памяти. А разные подвиды ассемблера применяются для программирования других устройств, где критичным является размер программы.

**С#**

[объектно-ориентированный](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%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) [язык программирования](http://ru.wikipedia.org/wiki/%D0%AF%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). Объе́ктно-ориенти́рованное программи́рование (ООП) — [парадигма программирования](http://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%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), в которой основными [концепциями](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%86%D0%B5%D0%BF%D1%86%D0%B8%D1%8F) являются понятия [объектов](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_(%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)) и [классов](http://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%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)). В случае [языков](http://ru.wikipedia.org/wiki/%D0%AF%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) с [прототипированием](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%BE%D0%B5_%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" \o "Прототипное программирование) вместо классов используются объекты-прототипы.

**Delphi**

[структурированный](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%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), [объектно-ориентированный](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%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) [язык программирования](http://ru.wikipedia.org/wiki/%D0%AF%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). Структу́рное программи́рование — методология разработки [программного обеспечения](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), в основе которой лежит представление программы в виде иерархической структуры [блоков](http://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%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%D0%B5)).

**SQL**

ЯП - это язык описания КОМПЬЮТЕРНЫХ ПРОГРАММ   
а SQL - язык структурированных запросов.

Стандартный SQL содержит в себе два языковых подмножества : Язык описания данных и Язык манипулирования данными. Data Definition Language Data Manipulation Language   
Первое подмножество предназначено для метаданных - для генерации таблиц, индексов, табличных пространств, пользователей, ролей, прав и т. д.   
Второе подмножество для запросов данных из созданных структур, удаления, добавления и модификации данных.

**Ruby**

[высокоуровневый язык программирования](http://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%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) для быстрого и удобного[[7]](http://ru.wikipedia.org/wiki/Ruby#cite_note-7)[[8]](http://ru.wikipedia.org/wiki/Ruby#cite_note-8) [объектно-ориентированного программирования](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%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). Язык обладает независимой от[операционной системы](http://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) реализацией [многопоточности](http://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C" \o "Многопоточность), строгой [динамической типизацией](http://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [сборщиком мусора](http://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0) и многими [другими возможностями](http://ru.wikipedia.org/wiki/Ruby#.D0.92.D0.BE.D0.B7.D0.BC.D0.BE.D0.B6.D0.BD.D0.BE.D1.81.D1.82.D0.B8_Ruby). Ruby близок по особенностям [синтаксиса](http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) к языкам [Perl](http://ru.wikipedia.org/wiki/Perl" \o "Perl) и [Eiffel](http://ru.wikipedia.org/wiki/Eiffel" \o "Eiffel), по объектно-ориентированному подходу — к [Smalltalk](http://ru.wikipedia.org/wiki/Smalltalk" \o "Smalltalk). Также некоторые черты языка взяты из [Python](http://ru.wikipedia.org/wiki/Python" \o "Python), [Lisp](http://ru.wikipedia.org/wiki/Lisp), [Dylan](http://ru.wikipedia.org/wiki/Dylan_(%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)" \o "Dylan (язык программирования)) и [Клу](http://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%83" \o "Клу). Динамический язык программирования с открытым исходным кодом с упором на простоту и продуктивность. Он обладает элегантным синтаксисом, который приятно читать и легко писать.

**UML**

унифицированный язык моделирования) — [язык](http://ru.wikipedia.org/wiki/%D0%AF%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) [графического](http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) описания для [объектного моделирования](http://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) в области [разработки программного обеспечения](http://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F). UML является языком широкого профиля, это — [открытый стандарт](http://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82), использующий графические обозначения для создания [абстрактной модели](http://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C) [системы](http://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), называемой*UML-моделью*. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML-моделей возможна [генерация кода](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%BE%D0%B3%D0%B5%D0%BD%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D1%8F).