Глава II

В данной главе, я хотела бы рассказать про три самые популярные классификации языков программирования. Почему же я, называя их, отношу в группу так называемых «популярных» языков программирования? Ответ очень прост, у них наиболее широкая область использования.

Давайте начнем с классификации по уровню. Как я уже сказала в первой главе, ее разделяют на две группы. В группу языков низкого уровня входят машинные языки и языки символического кодирования[[1]](#footnote-1): Автокод, Ассемблер. Операторы данного языка – это те же машинные команды, но записанные мнемоническими кодами[[2]](#footnote-2), а в качестве операндов[[3]](#footnote-3) употребляются не точные адреса, а символические имена. Все языки низкого уровня ориентированы на определенный тип компьютера, т. е. являются машинно – зависимыми. Машинно – ориентированные языки – это языки, наборы операторов и изобразительные средства которых значительно зависят от отличительных черт ЭВМ (внутреннего языка, структуры памяти и т.д.). Теперь про языки символического кодирования.

Ассемблер. Свое название данный язык приобрел от имени системной программы Ассемблер, которая преобразует изначальные программы, написанные на таких языках, именно в коды машинных команд. Частями тут служат операторы, а в результате сборки получается последовательность машинных команд. Язык Ассемблера соединяет в себе плюсы языка машинных команд и некоторые черты языков высокого уровня. Ассемблер обеспечивает вероятность применения символических имен в начальной программе и избавляет программиста от утомительного труда по распределению памяти компьютера для команд, переменных и констант.

Ассемблер кроме того гибко и полно использует технические возможности компьютера, как и язык машинных команд. Транслятор начальных программ в Ассемблере легче транслятора, требующегося для языка программирования высокого уровня. На Ассемблере можно написать столь же успешную по размеру и времени выполнения программу, как и программу на языке машинных команд. Данное преимущество отсутствует у языков высокого уровня. Данный язык нередко применяют для программирования систем реального времени, технологическими процессами и оборудованием, обеспечение работы информационно-измерительных комплексов. К таким системам обычно предъявляются высокие требования по объему занимаемой машинной памяти. Нередко язык Ассемблера дополняется средствами формирования макрокоманд, каждая из которых эквивалентна целой группе машинных команд. Такой язык называют языком макроассемблера. Использование мак "строительных" блоков и приближает язык Ассемблера к языку высокого уровня. Ассемблер машинно-зависимый язык, т. е. он отражает отличительные черты архитектуры конкретного типа компьютера. [[2]](http://bourabai.ru/alg/classification04.htm)

Что же такое транслятор? Транслятор – специальная программа, предназначенная для перевода программы с языка программирования на язык машинных кодов. Например, программу, написанную на языке программирования высокого уровня, ЭВМ не понимает, поскольку ей доступен только машинный язык. И чтобы язык высокого уровня был понятен, и используют транслятор. [[4]](http://studopedia.ru/3_201967_algoritmicheskie-yaziki-nizkogo-i-visokogo-urovnya.html)

Существует три вида транслятора: интерпретаторы, компиляторы и ассемблеры.

Чем же различаются данные виды транслятора?

Интерпретатор – это транслятор, который производит пооператорную обработку и выполнение исходного кода программы. [[4]](http://studopedia.ru/3_201967_algoritmicheskie-yaziki-nizkogo-i-visokogo-urovnya.html)

Компилятор – транслятор, преобразующий всю программу в модуль на машинном языке, после чего программа записывается в память компьютера и лишь потом исполняется. [[4]](http://studopedia.ru/3_201967_algoritmicheskie-yaziki-nizkogo-i-visokogo-urovnya.html)

Ассемблер – это транслятор, переводящий программу, записанную на языке Ассемблера, в программу на машинном языке. Самое главное не путать эти два понятия: язык Ассемблера и данный транслятор. [[4]](http://studopedia.ru/3_201967_algoritmicheskie-yaziki-nizkogo-i-visokogo-urovnya.html)

Второй язык символического кодирования:

Автокод — язык программирования, предложения которого по своей структуре в основном подобны командам и обрабатываемым данным конкретного машинного языка. Автокод – это как дополнение Ассемблера. [[3]](http://translate.academic.ru/%D0%90%D0%92%D0%A2%D0%9E%D0%9A%D0%9E%D0%94/ru/)

Также к языкам низкого уровня относятся: Java и Microsoft.net. Java — строго типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java – машины. Язык Java очень сильно связан с Internet, потому, что первой серьезной программой, которая была написана на данном языке, была браузером Всемирной паутины. [[5]](http://javastudy.ru/java-core/)

В последнее время, говоря о программировании в Internet, часто имеют в виду создание публикаций с использованием языка HTML. Применение специальных средств, к примеру, HTML – редакторов, позволяет не только создавать отдельные динамически изменяющиеся интерактивные HTML – документы, используя при этом данные мультимедиа, но и редактировать целые сайты. [[10]](http://life-prog.ru/1_21301_instrumentalnie-sredstva-programmirovaniya.html)

Наверное, сейчас уже не существует как такого языка Microsoft.net. Есть некая программная платформа Microsoft.net, которая была выпущена в 2002 году компанией Microsoft. Основой платформы является общеязыковая среда исполнения Common Language Runtime (CLR), которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду. Одной из основных идей Microsoft.net является совместимость программных частей, написанных на разных языках. Например, в Microsoft используются такие языки как: Visual Basic.NET; Jscript.NET; C++/CLI – новая версия Managed C++; F#; J#; C#. [[11]](http://needworld.ru/threads/microsoft-net-framework.10542/)

К языкам программирования высокого уровня относят Фортран, Алгол, Кобол, Паскаль, Бейсик, Си, Пролог и другие.

Фортран – первый реализованный язык программирования высокого уровня (после Планкалкюля), правда, с одной небольшой оговоркой — для машин, построенных по классической схеме фон Неймана. Данный язык был создан в период с 1954 по 1959 года. Руководителем группы программистов по созданию данного языка был Джона Бэкуса. [[7]](http://progopedia.ru/language/fortran/)

Кобол – коммерческий язык, который в основном используется для программирования экономических задач. Первая версия языка была выпущена в 1959 году. Руководителем проекта по созданию данного языка был Грейс Хоппер. [[8]](http://www.ngpedia.ru/id87879p2.html)

Пролог – язык, в основе которого лежит аппарат математической логики. Был разработан в 1972 году Аленом Колмероэ. [[9]](http://labs.org.ru/prolog-1/)

Эти языки машинно–независимы, так как они ориентированы на систему операндов, характерных для записи определенного типа алгоритмов. Но программы, которые написаны на языках высокого уровня, занимают больше памяти и медленнее выполняются, чем программы на машинных языках.

Например, в нашей школе основу программирования начинают проходить с помощью языка высокого уровня – Pascal. Можно немного отвлечься от темы и рассказать вам про него. Данный язык наиболее распространен среди учащихся старших классов и студентов первых курсов вуза, ведь он является базой для изучения большинства других языков программирования. Pascal назван в честь Блеза Паскаля. Это чисто процедурный язык программирования, и как я уже сказала, он часто используется для обучения структурному программированию. Данный язык относится к классификации по уровню, и принадлежит группе языков высокого уровня. [[11]](http://bourabai.ru/alg/pascal/index.html)

Классификация языков программирования по поколениям.

– языки первого поколения: машинно–ориентированные с ручным управлением памяти на компьютерах первого поколения.

К первому поколению относят машинные языки — языки программирования на уровне команд процессора конкретной машины. Для программирования не используется транслятор, команды вводятся непосредственно в машинном коде переключателями на передней панели машины. Такие языки хорошо подходят для детального понимания функционирования конкретной машины, но сложны для изучения и решения прикладных задач.

– языки второго поколения: языки с мнемоническим представлением команд, так называемые автокоды.

Языки второго поколения создавались для того, чтобы облегчить тяжёлую работу по программированию, перейдя в выражениях языка от низкоуровневых машинных понятий ближе к тому, как по правилу мыслит программист. Эти языки появились в 1950-е годы, в частности, такие языки как Фортран и Алгол. Наиболее значимой проблемой, с которой столкнулись разработчики языков второго поколения, стала задача убедить клиентов в том, что созданный компилятором код выполняется достаточно хорошо, чтобы оправдать отказ от программирования на ассемблере. Скептицизм по поводу возможности создания эффективных программ с помощью автоматических компиляторов был довольно распространён, поэтому разработчикам таких систем должны были продемонстрировать, что они действительно могут генерировать почти такой же эффективный код, как и при ручном кодировании, причём практически для любой исходной задачи.

– языки третьего поколения: общего назначения, используемые для создания прикладных программ любого типа. Например, Бейсик, Кобол, Си и Паскаль. В принципе можно сказать, что большинство языков третьего поколения – языки высокого уровня.

Под третьим поколением первоначально понимались все языки уровнем выше, чем Ассемблер. Главной отличительной чертой языков третьего поколения стала независимость от аппаратного обеспечения, то есть выражение алгоритма в форме, не зависящей от конкретных характеристик машины, на которой он будет исполняться. Код, который написан на языке третьего поколения, перед исполнением транслируется либо непосредственно в машинные команды, либо в код на ассемблере и затем уже ассемблируется. При компиляции, в отличие от предыдущих поколений, уже нет соответствия один-к-одному между инструкциями программы и генерируемым кодом.

– языки четвертого поколения: усовершенствованные, разработанные для создания специальных прикладных программ, для управления базами данных.

Термин языки программирования четвёртого поколения лучше представлять, как среды разработки четвёртого поколения. Они относятся к временному периоду с 1970-х по начало 1990-х.

Языки этого поколения предназначены для реализации крупных проектов, повышают их надежность и скорость создания, ориентированы на специализированные области применения, и используют не универсальные, а объектно-ориентированные языки, оперирующие конкретными понятиями узкой предметной области. В эти языки встраиваются мощные операторы, позволяющие одной строкой описать такую функциональность, для реализации которой на языках младших поколений потребовались бы тысячи строк исходного кода.

– языки программирования пятого поколения: языки декларативные, объектно–ориентированные и визуальные. Например, Пролог, ЛИСП, Си++, Visual Basic, Delphi.

ЛИСП – язык, который используется для построения программ с использованием методов искусственного интеллекта.

Рождение языков пятого поколения произошло в середине 90-х годов. К ним относятся также системы автоматического создания прикладных программ с помощью визуальных средств разработки, без знания программирования. Основная идея была заключена в возможности автоматического формирования результирующего текста на универсальных языках программирования. Инструкции же вводятся в компьютер в максимально наглядном виде с помощью методов, наиболее удобных для человека, не знакомого с программированием.

Теперь давай те поподробнее поговорим про классификацию по направленности. В процедурных языках программа описывает действия, которые необходимо выполнить, а результат задается только способом получения его при помощи некоторой процедуры, представляющей из себя определенную последовательность действий.

Среди процедурных языков выделяют в свою очередь структурные и операционные языки. В структурных языках одним оператором записываются целые алгоритмические структуры: ветвления, циклы и т.д. В операционных языках для этого используются несколько операций. Вот примеры структурных языков, которые наиболее распространены на данный момент: Паскаль, Си, Ада, ПЛ/1. Среди операционных известны Фортран, Бейсик, Фокал.

Непроцедурное (декларативное) программирование появилось в начале семидесятых годов XX века. К непроцедурному программированию относятся функциональные и логические языки.

В функциональных языках программа описывает вычисление некоторой функции. Обычно эта функция задается как композиция других, более простых, те в свою очередь делятся на еще более простые задачи и т.д. Один из основных элементов функциональных языков – рекурсия. Оператора присваивания и циклов в классических функциональных языках нет.

В логических языках программа вообще не описывает действий. Она задает данные и соотношения между ними. После этого системе можно задавать вопросы. Машина перебирает известные и заданные в программе данные и находит ответ на вопрос. Порядок перебора не описывается в программе, а неявно задается самим языком. Классическим языком логического программирования считается Пролог. Программа на Прологе содержит, набор предикатов–утверждений, образующие проблемно–ориентированную базу данных и правила, которые имеют вид условий.

1. Язык символического кодирования – язык, представляющий собой систему операторов, соответствующих понятию машинной команды, в которых используются мнемонические обозначения кодов операций и символические названия операндов и элементов программы. [↑](#footnote-ref-1)
2. Мнемонический код – это условное символическое обозначение оператора, содержащее до пяти символов. [↑](#footnote-ref-2)
3. Операнд – величина, представляющая собой объект операции, реализуемой ЭВМ в ходе выполнения программы вычислений. [↑](#footnote-ref-3)