## § 1. ИСТОРИЯ РАЗВИТИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

ХУЙ

Одной из самых революционных идей, приведших к созданию автоматических цифровых вычислительных машин, была высказанная в 20-х годах XIX века Ч.Бэббиджем мысль о предварительной записи порядка действий машины для последующей автоматической реализации вычислений - программе. И, хотя использованная Бэббиджем запись программы на перфокартах, придуманная для управления ткацкими станками французским изобретателем Жозефом Мари-Жаккаром, технически не имеет ничего общего с современными приемами хранения программ в ЭВМ, принцип здесь по-существу один. С этого момента начинается история программирования.

Аду Лавлейс, одну из немногих современников Чарльза Бэббиджа, кто сумел по достоинству оценить аналитическую машину, называют первым в мире программистом. Она теоретически разработала некоторые приемы управления последовательностью вычислении, которые используются в программировании и по сей день, описала одну из важнейших конструкций практически любого современного языка программирования - цикл.

Революционным моментом в истории языков программирования стало появление системы кодирования машинных команд с помощью специальных символов, предложенной Джоном Моучли, сотрудником Пенсильванского университета. Система кодирования, предложенная Моучли, увлекла одну из сотрудниц его компании - Грейс Мюррей Хоппер, которая посвятила всю свою жизнь компьютерам и программированию. Она вспоминает, что стала «третьим в мире программистом первого в мире большого цифрового компьютера». Г.Хоппер доказала, чего она стоит как программист. Впоследствии она писала: «Я имела то преимущество, что изучала как технику, так и математику и знала, как работает машина от начала и до конца».

При работе на компьютере «Марк-1» Г.Хоппер и ее группе пришлось столкнуться со многими проблемами и все, что ими придумано, было впервые. В частности, они придумали подпрограммы. Сейчас любой программист не задумываясь использует аппарат подпрограмм в любом языке программирования. И еще одно фундаментальное понятие техники программирования впервые ввели Г.Хоппер и ее группа - «отладка». Однажды жарким летним днем 1945 г. неожиданно произошла остановка компьютера «Марк-1». Обнаружилась неисправность одного реле, контакты которого были заблокированы мотыльком, залетевшим неизвестно каким образом в помещение вычислительного центра. Вспоминает Г.Хоппер: «Когда к нам зашел офицер, чтобы узнать, чем мы занимаемся, мы ответили, что очищаем компьютер от насекомых (debuging)». С тех пор термин «debuging» (отладка) используется в технических процессах тестирования неисправностей в компьютере, а также в системах программирования.

На заре компьютерной эры машинный код был единственным средством общения человека с компьютером. Огромным достижением создателей языков программирования было то, что они сумели заставить сам компьютер работать переводчиком с этих языков на машинный код.

В конце 40-х годов, до прихода Г.Хоппер в фирму Джона Моучли, последний создал систему под названием «Short Code», которая являлась примитивным языком программирования высокого уровня. В ней программист записывал решаемую задачу в виде математических формул, а затем, используя специальную таблицу, переводил символ за символом, преобразовывал эти формулы в двухлитерные коды. В дальнейшем специальная программа компьютера превращала эти коды в двоичный машинный код. Система, разработанная Дж. Моучли, была по существу одним из первых примитивных интерпретаторов.

Уже в 1951 г. Хоппер создала первый в мире компилятор и ею же был введен сам этот термин. Компилятор Хоппер осуществлял функцию объединения команд и в ходе трансляции производил организацию подпрограмм, выделение памяти компьютера, преобразование команд высокого уровня (в то время псевдокодов) в машинные команды. «Подпрограммы находятся в библиотеке (компьютера), а когда вы подбираете материал из библиотеки - это называется компиляцией» - так она объясняла происхождение введенного ею термина.

В 1954 г. группа под руководством Г.Хоппер разработала систему, включающую язык программирования и компилятор, которая в дальнейшем получила название MATH-MATIC. После удачного завершения работ по созданию MATH-MATIC Г.Хоппер и ее группа принялись за разработку нового языка и компилятора, который позволил бы пользователям программировать на языке, близком к обычному английскому. Необходимость появления подобной системы Хоппер объясняла следующим образом: «Существует много различных людей, которым нужно решать разные задачи. Некоторые из них связаны с обработкой символов, другие - с обработкой слов, и им нужны языки другого типа, а не наши попытки превратить их всех в математиков». В 1958 г. появился компилятор FLOW-MATIC. В отличие от ФОР-ТРАНа - языка для научных приложений - FLOW-MATIC был первым языком для задач обработки коммерческих данных. Работы в этом направлении привели к созданию языка КОБОЛ (COBOL - Common Business Oriented Language). Одним из основных консультантов при создании этого языка была Грейс Мюррей Хоппер.

Середина 50-х годов характеризуется стремительным прогрессом в области программирования. Роль программирования в машинных командах стала уменьшаться. Начали появляться языки программирования нового типа, выступающие в роли посредника между машинами и программистами. Первым и одним из наиболее распространенных был Фортран (FORTRAN, от FORmula TRANslator - переводчик формул), разработанный группой программистов фирмы IBM в 1954 г. (первая версия).

В середине 60-х годов сотрудники математического факультета Дартмутского колледжа Томас Курц и Джон Кемени создали специализированный язык программирования, который состоял из простых слов английского языка. Новый язык назвали «универсальным символическим кодом для начинающих» (Beginners All-Purpose Symbolic Instruction Code, или, сокращенно, BASIC). Годом рождения нового языка можно считать 1964 г. Сегодня универсальный язык Бейсик (имеющий множество версий) приобрел большую популярность и получил широкое распространение среди пользователей ЭВМ различных категорий во всем мире. В значительной мере этому способствовало то, что Бейсик начали использовать как встроенный язык персональных компьютеров, широкое распространение которых началось в конце 70-х годов.

В начале 60-х годов все существующие языки программирования высокого уровня можно было пересчитать по пальцам, однако впоследствии их число достигло трех тысяч. Разумеется, подавляющая часть языков не получила сколько-нибудь широкого распространения; в практической деятельности используется не более двух десятков. Разработчики ориентировали языки на разные классы задач, в той или иной мере привязывали их к конкретным архитектурам ЭВМ, реализовывали личные вкусы и идеи. В 60-е годы были предприняты попытки преодолеть эту «разноголосицу» путем создания универсального языка программирования. Первым детищем этого направления стал PL/I (Programm Language One), 1967 г. Затем на эту роль претендовал АЛГОЛ-68 (1968 г.). Предполагалось, что подобные языки будут развиваться и усовершенствоваться и вытеснят все остальные. Однако ни одна из этих попыток на сегодняшний день не увенчалась успехом (хотя PL/I в усеченных версиях использовали многие программисты). Всеохватность языка приводила к неоправданной, с точки зрения программиста, сложности конструкций, неэффективности компиляторов.

Языки программирования служат разным целям иих выбор определяется удобностью пользователя, пригодностью для данного компьютера и данной задачи. А задачи для компьютера бывают самые разнообразные: вычислительные, экономические, графические, экспертные и т.д. Такая разнотипность решаемых компьютером задач и определяет многообразие языков программирования. По всей видимости, в программировании наилучший результат достигается при индивидуальном подходе, исходящем из. класса задачи, уровня и интересов программиста. Например, Бейсик широко употребляется при написании простых программ; Фортран является классическим языком программирования при решении на ЭВМ математических и инженерных задач; язык Кобол (COBOL, от Common Business Oriented Language -общий язык, ориентированный на деловые задачи; создан в 1960 г.) был задуман как основной язык для массовой обработки данных в сферах управления и бизнеса. Еще более специализированным является язык ЛОГО (от греческого logos - слово), созданный для обучения программированию школьников профессором математики и педагогики Сеймуром Пейпертом из Массачусетского технологического института. Обучаясь программированию на ЛОГО, дети задают простые команды, которые управляют игрушечной черепашкой, снабженной карандашиком. Отметим и еще один достаточно популярный специализированный язык - Пролог (Prolog -PROgramming in LOGic), разработанный как язык программирования для создания систем искусственного интеллекта.

В конце 50-х годов плодом международного сотрудничества в области программирования явился Алгол (ALGOL, от ALGOrithmic Language - алгоритмический язык). Алгол предназначен для записи алгоритмов, которые строятся в виде последовательности процедур, применяемых для решения поставленных задач. Специалисты-практики восприняли этот язык далеко неоднозначно, но, тем не менее, его влияние на развитие других языков и теорию программирования оказалось весьма значительным.

В нашей странев те годы был создан под руководством Сергея Петровича Ершова транслятор Альфа, который представлял довольно удачную русифицированную версию Алгола. Впоследствии академик Ершов сыграл важнейшую роль в становлении в СССР школьной информатики.

Развитие идеи Алгола о структуризации разработки алгоритмов нашло наивысшее отражение при создании в начале 70-х годов языка Паскаль швейцарским ученым Никлаусом Виртом. Язык Паскаль первоначально разрабатывался как учебный, и, действительно, сейчас он является одним из основных языков обучения программированию в школах и вузах. Однако, качества его в совокупности оказались столь высоки, что им охотно пользуются и профессиональные программисты.

Не менее впечатляющей, в том числе и финансовой, удачи добился джазист Филип Кан, француз, разработавший систему Турбо-Паскаль. Суть его идеи состояла в объединении последовательных этапов обработки программы - компиляции, редактирования связей, отладки и диагностики ошибок - в едином интерфейсе. Версии Турбо-Паскаля заполонили практически все образовательные учреждения, программистские центры и частные фирмы.

Период с конца 60-х и до начала 80-х годов характеризуется бурным ростом числа различных языков программирования, сопровождавшим, как это ни парадоксально, крмзис программного обеспечения. Этот кризис особо остро переживало военное ведомство США. В январе 1975 г. Пентагон решил навести порядок в хаосе трансляторов и учредит комитет, которому было предписано разработать один универсальный язык. На конкурсной основе комитет проработал сотни проектов и, когда стало ясно, что ни один из существующих языков не может их удовлетворить, принял два проекта для окончательного рассмотрения. В мае 1979 г. был объявлен победитeль - группа ученых во главе с Жаном Ихбиа. Победивший язык окрестили АДА, в честь Огасты Ады Лавлейс. Язык АДА - прямой наследник языка Паскаль. *-* этот язык предназначен для создания и длительного (многолетнего) сопровождения больших программных систем, допускает возможность параллельной обработки, правления процессами в реальном времени и многое другое, чего трудноилиневозможно достичь средствами более простых языков.

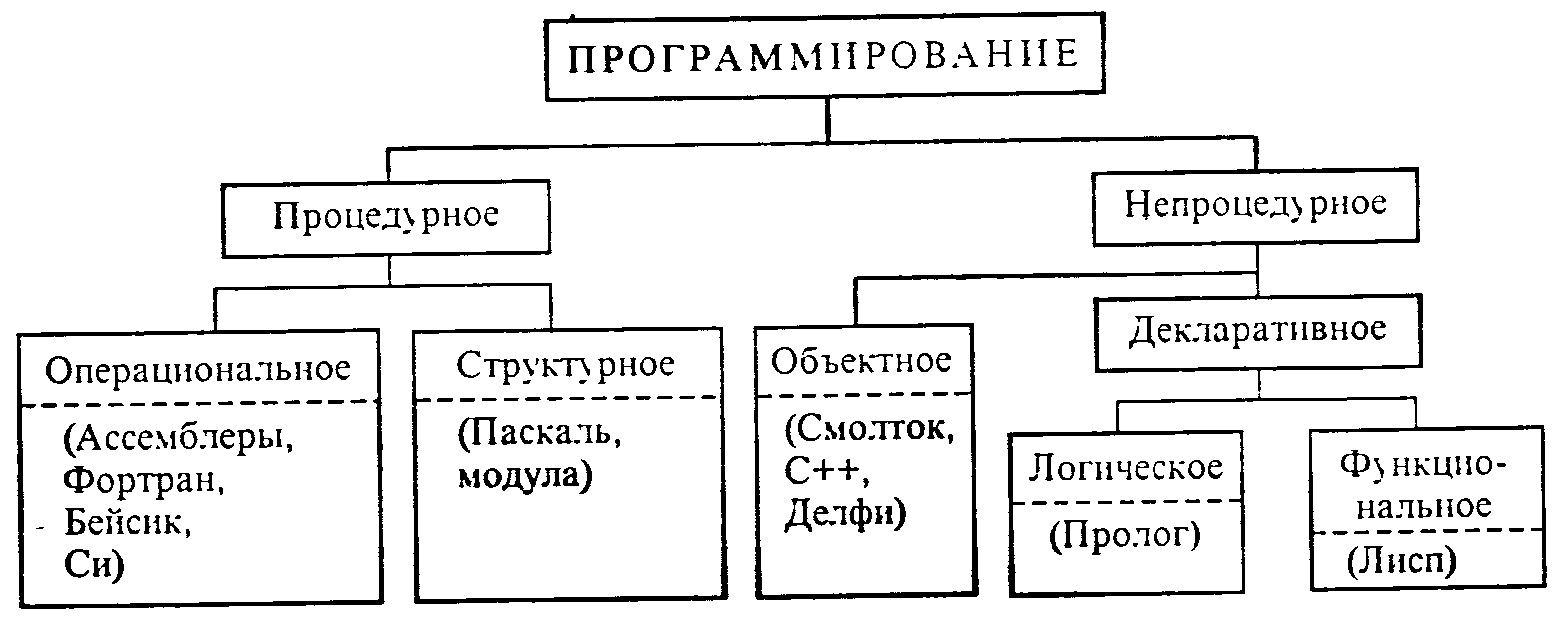
Большой отпечаток на современное программирование наложил язык Си (первая версия - 1972 г.), являющийся очень популярным в среде разработчиков систем программного обеспечения (включая операционные системы). Си сочетает в себе черты как языка высокого уровня, так и машинно-ориентированного языка, допуская программиста ко всем машинным ресурсам, чего не обеспечивают такие языки, как Бейсик и Паскаль.

Следует отметить, что многие языки, первоначально разработанные для больших и малых ЭВМ, в дальнейшем были приспособлены к персональным компьютерам. Хорошо вписались в «персоналки» не только Паскаль, Бейсик, Си, Лого, но и ЛИСП, ПРОЛОГ - языки искусственного интеллекта.

В течение многих лет программное обеспечение строилось на основе операциональных и процедурных языков, таких как Фортран, Бейсик, Паскаль, Ада, Си. И сегодня современные версии этих и им подобных языков (Модула, Форт и др.) доминируют при разработке прикладных программных средств. Однако по мере эволюции языков программирования получили широкое распространение и другие, принципиально иные, подходы к созданию программ.

Классическое операциональное и/или процедурное программирование требует от программиста детального описания того, как решать задачу, т.е. формулировки алгоритма и его специальной записи. При этом ожидаемые свойства результата обычно не указываются. Основные понятия языков этих групп - оператор и данные. При процедурном подходе операторы объединяются в группы - процедуры. Структурное программирование в целом не выходит за рамки этого направления, оно лишь дополнительно фиксирует некоторые полезные приемы технологии программирования.

Принципиально иное направление в программировании связано с методологиями (иногда говорят «парадигмами») непроцедурного программирования. К ним можно отнести объектно-ориентированное и декларативное программирование. Объектно-ориентированный язык создает окружение в виде множества независимых объектов. Каждый объект ведет себя подобно отдельному компьютеру,их можно использовать для решения задач как «черные ящики», не вникая во внутренние механизмы их функционирования. Из языков объектного программирования, популярных среди профессионалов, следует назвать прежде всего Си++, для более широкого круга программистов предпочтительны среды типа Delphi и Visual Basic.



*Рис. 3.1.* Классификация языков программирования

При использовании декларативного языка программист указывает исходные информационные структуры, взаимосвязи между ними и то, какими свойствами должен обладать результат. При этом процедуру его получения («алгоритм») программист не строит (по крайней мере, в идеале). В этих языках отсутствует понятие «оператор» («команда»). Декларативные языки можно подразделить на два семейства - логические (типичный представитель - Пролог) и функциональные (Лисп). По всей видимости, непроцедурные языки имеют большое будущее. Все сказанное выше можно отобразить схемой - крупноструктурной классификацией языков программирования. В ней указаны основные методологии программирования; в нижнем ряду, в скобках - типичные языки соответствующих групп.