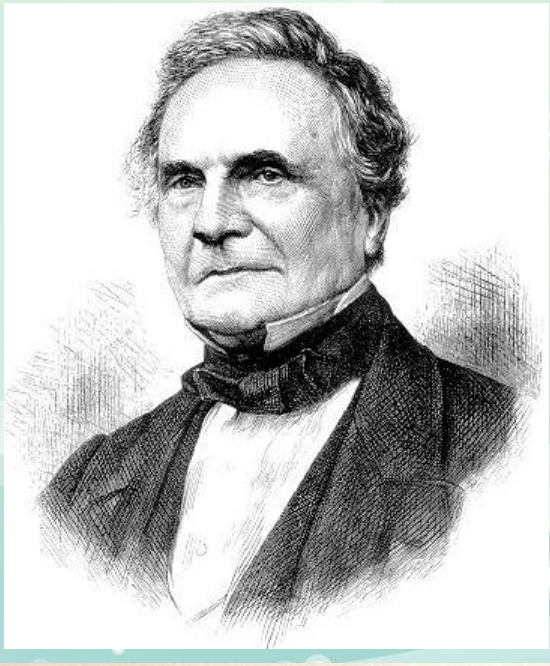
История развития языков программирования

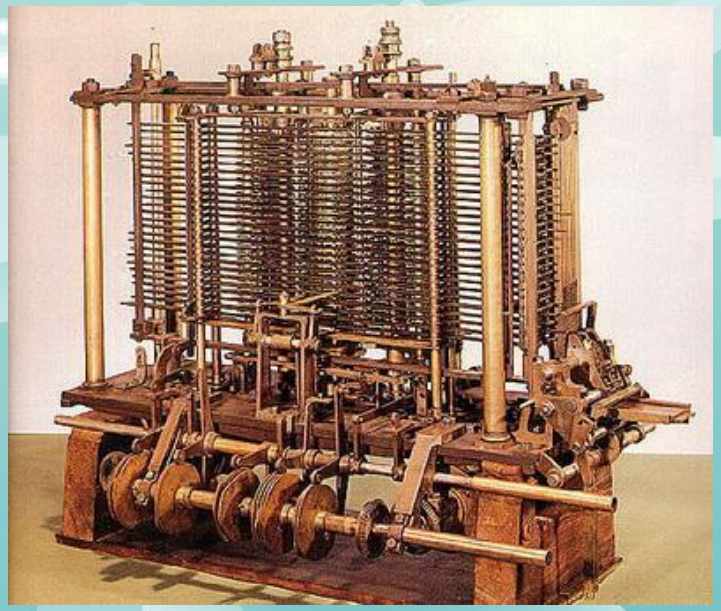
Выполнила: Красова Виктория

Содежание

Ча́рльз Бэ́ббидж

Разностная машина - механический аппарат для автоматизации вычислений путём аппроксимации функций многочленами и вычисления конечных разностей.





Первая программистка

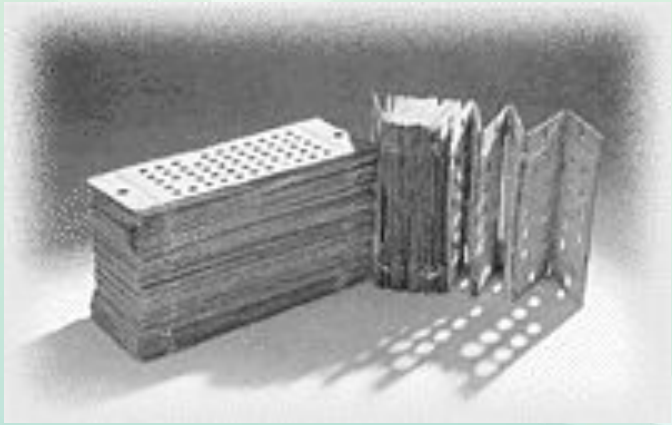
Августа Ада Кинг (урождённая Байрон), графиня Лавлейс

Составила первую в мире программу (для **Аналитической машины** Чарльза Бэббиджа).

Аналитическая машина Бэббиджа должна была производить разнообразные вычисления, следуя набору инструкций.

Августа Ада Байрон (графиня Лавлейс) 1815 – 1852 дочь поэта Джорджа Байрона). Она была ни на кого не похожа и обладала талантом не поэтическим, метафизическим. Наряду с совершенно мужской способностью к пониманию, проявляющейся в умении решительно и быстро схватывать суть дела в целом, леди Лавлейс обладала всеми прелестями утонченного женского характера. Ее манеры ее вкусы, ее образование, особенно музыкальное, в котором она достигла совершенства, были женственными в наиболее прекрасном смысле этого слова, и поверхностный наблюдатель никогда не угадал бы, сколько внутренней силы и знания сокрыто под ее женской грацией. Ада Лавлейс – ученица и помощница Чарльза Бэббиджа. Основные составляющие вклада в "вычислительную науку": - введение понятий цикл и рабочая ячейка; - идея программного управления процессом вычисления; - изобретение системы для ускорения расчетов; - использование перфокарт для ввода и вывода информации. Графиня Лавлейс вошла в историю компьютерной техники как первый программист, ее именем назван язык программирования Ада.

Августа Ада Байрон



Именно Лавлейс принадлежит идея использования для подачи на вход машины двух потоков перфокарт, которые были названы операционными картами и картами переменных: первые управляли процессом обработки данных, которые были записаны на вторых.

Информация заносилась на перфокарты путем пробивки отверстий. Из операционных карт можно было составить библиотеку функций. Помимо этого, Analytical Engine, по замыслу автора, должна была содержать устройство печати и устройство вывода результатов на перфокарты для последующего использования.

При проектировании Аналитической машины в 1836-1848 годах Бэббидж фактически задал направление всему последующему развитию ЭВМ. Проект создания аналитической машины предусматривал целый ряд механизмов, присущих нынешним ЭВМ:

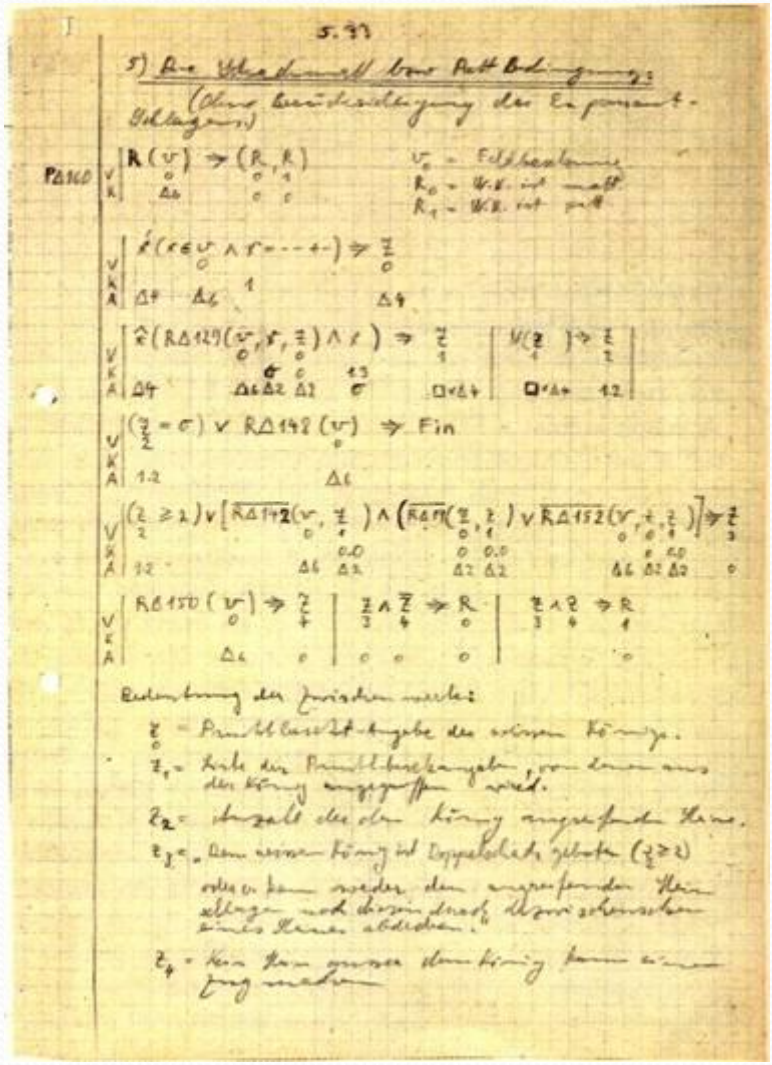
1. Тех же пяти компонент (арифметическое устройство, устройство памяти, управления, ввода и вывода)
2. В число операций, помимо четырех арифметических, была включена операция условного перехода и операции с кодами команд
3. Все программы вычислений записывались на перфокартах пробивками

ПРИЧИНЫ НЕУДАЧИ БЭББИДЖА

Основная причина: Бэббидж действительно слишком превзошел свое время (в конце жизни он сказал: «я готов отдать последние годы своей жизни за то, чтобы прожить три дня через 150 лет, и чтобы мне подробно объяснили принцип работы будущих машин»). Бэббидж не сомневался в будущем развитии вычислительной техники.

- **Невозможность** в то время **обрабатывать металл с высокой степенью точности** (в то время как для реализации проекта Аналитической машины только зубчатых колес потребовалось бы несколько тысяч!)
- **Финансовая проблема**. Если поначалу различные научные общества с энтузиазмом поддерживали Бэббиджа, то совсем скоро они охладели к затратному проекту с размытыми целями. В 1851 году Бэббидж с горечью заявлял, что все, связанное с машиной, он сделал за собственные деньги. Известно, что ученый в целях добычи материальных средств написал роман, пытался избраться в Парламент Британской империи, даже одно время играл в лотерею.

1940-ые, Конрад Цузе, Plancalcul



Первая попытка создать высоко-уровневый язык программирования принадлежит гениальному Конраду Цузе (конец 1940-х годов), разработавшему Plancalcul (планировщик вычислений).

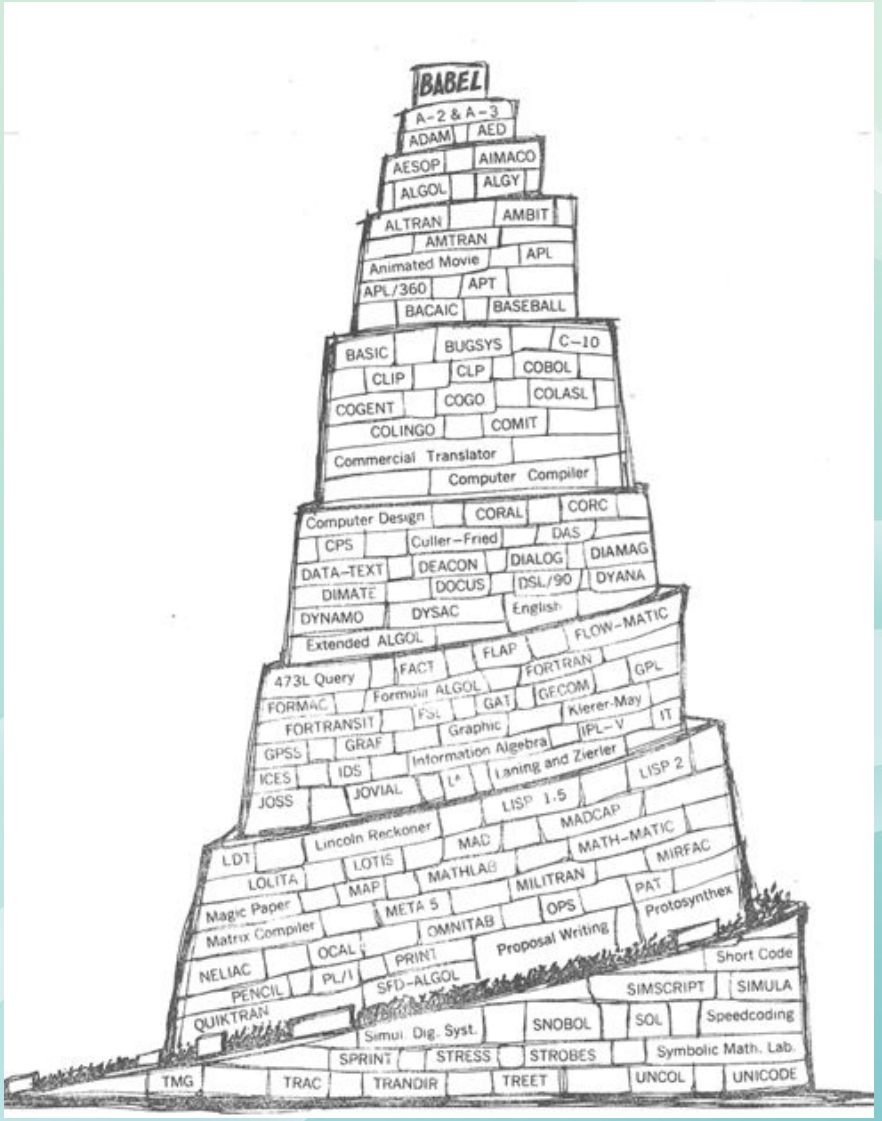
«Plancalcul родился исключительно как результат теоретической работы, без всякой связи с тем, появится или нет в обозримом будущем машины, подходящие к программам на Plancalcul».

Фрагмент рукописи Цузе с шахматной программой на языке Plancalcul

Языки программирования в СССР

Михаил Романович Шура-Бура и А.П. Ершов – создатели первых отечественных систем автоматизации программирования для ЭВМ «БЭСМ» и «Стрела»
(1954-1956 годы)

Языки и системы программирования в 1960-е

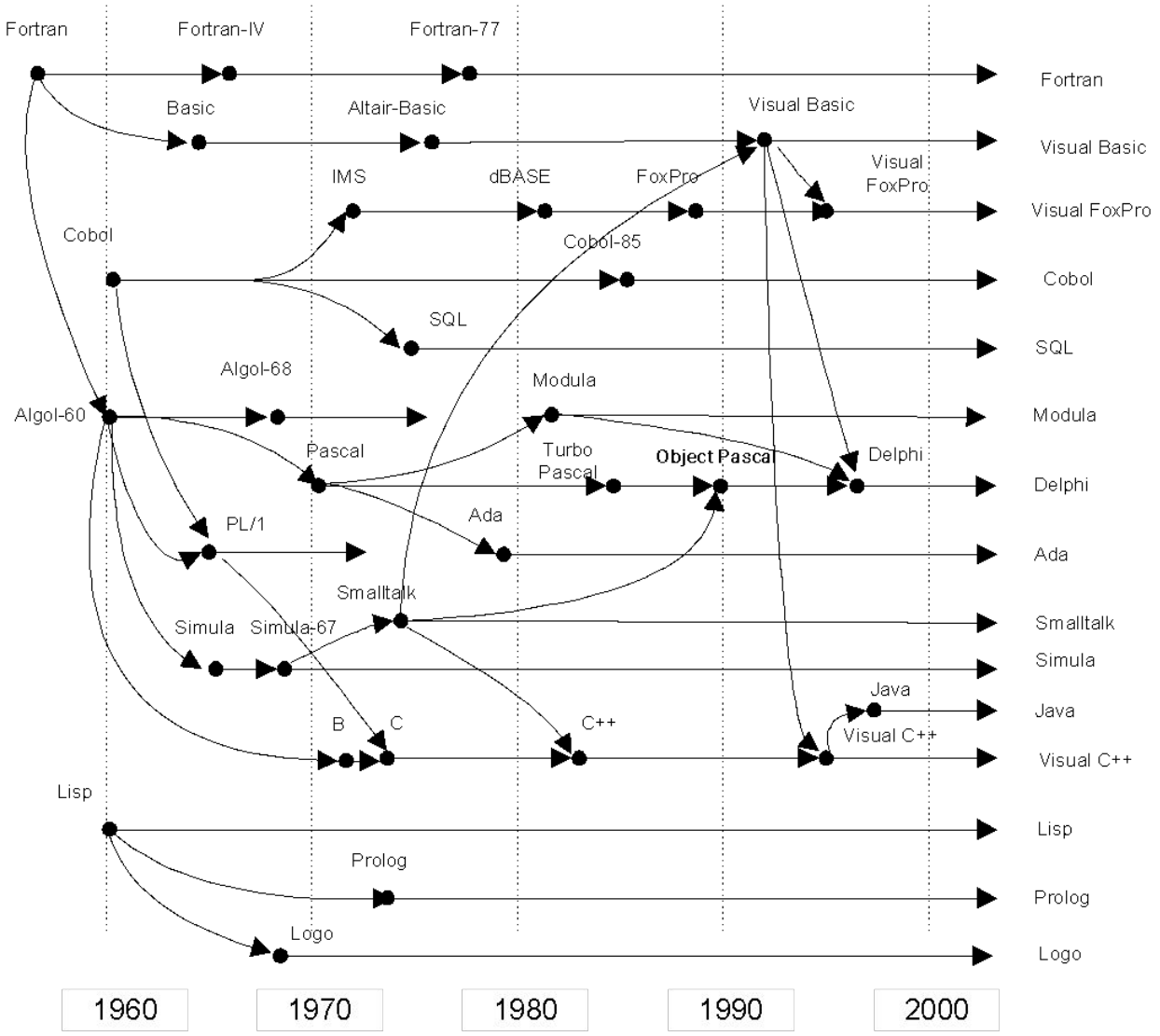


Наиболее активный период разработки языков и систем программирования приходится на 1960-е годы.

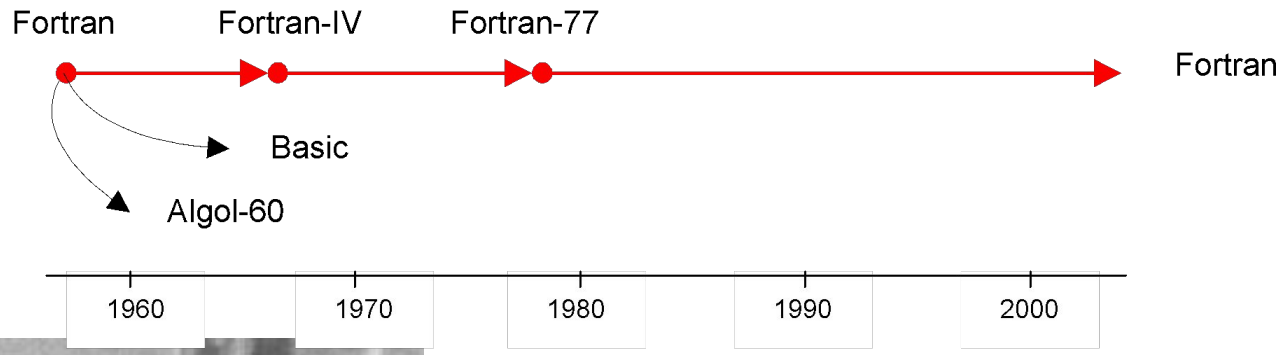
За это десятилетие в мире родилось более тысячи разнообразных языков, как универсальных, так и специализированных, но выжили и доросли до XXI века дожили немногие, в том числе бессмертные Fotran, Basic, Algol, Cobol, Simula, Lisp и их потомки.

На рисунке: «вавилонская башня» языков программирования, созданных в 1960-е годы

Родословная основных высокоуровневых языков программирования



Бессмертный Fortran

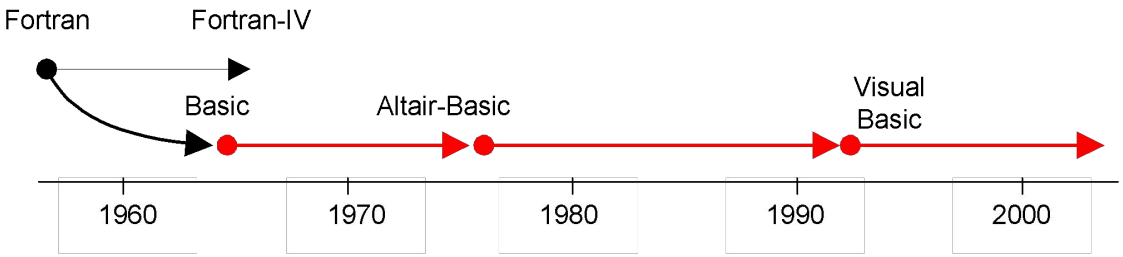


Fortran = FORmula TRANslator

Первый высокоуровневый язык программирования Fortran был разработан в фирме IBM под руководством Джона Бэкуса (Backus, John; p. 1924).

Работа над языком началась в 1954 г., первая реализация для IBM 704 в выполнена в 1957 г.

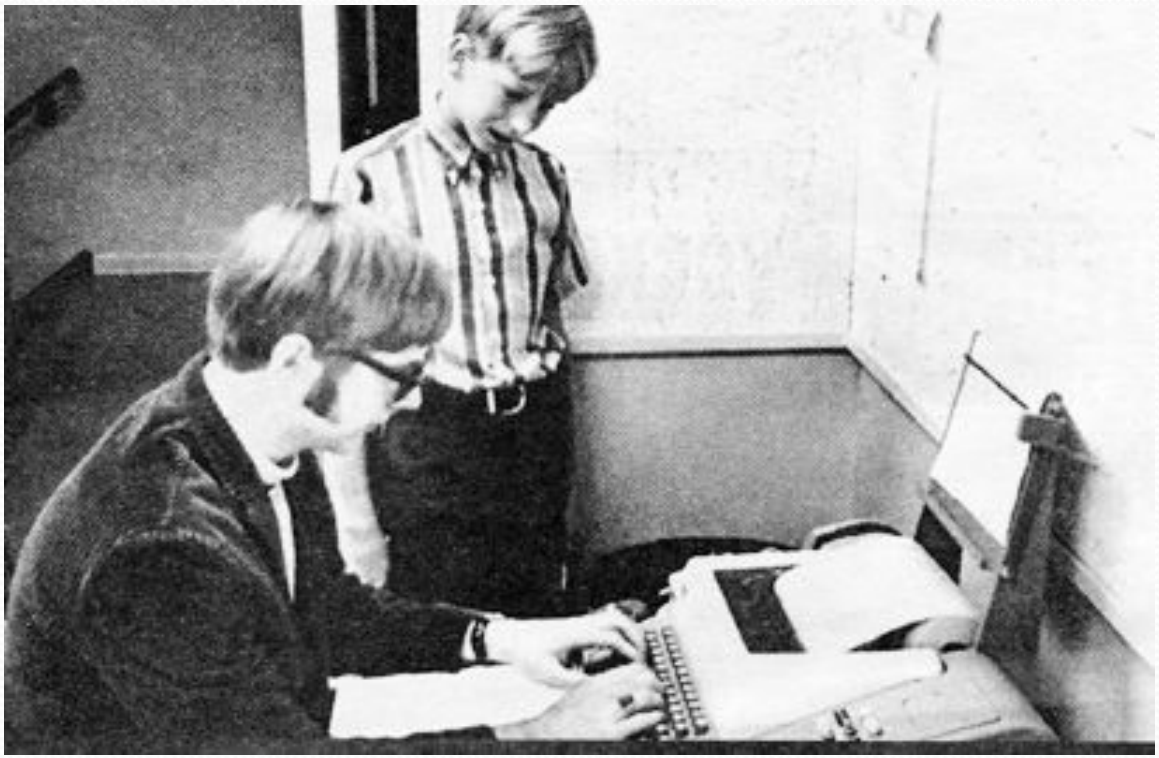
Basic – язык для начинающих



BASIC = Beginners All-purpuse Symbolic Instruction Code

Язык Basic был разработан в 1964 г. в Дармутском колледже в г. Хановере (Darmouth College, Hanover), штат Нью-Хемпшир

Basic и Microsoft

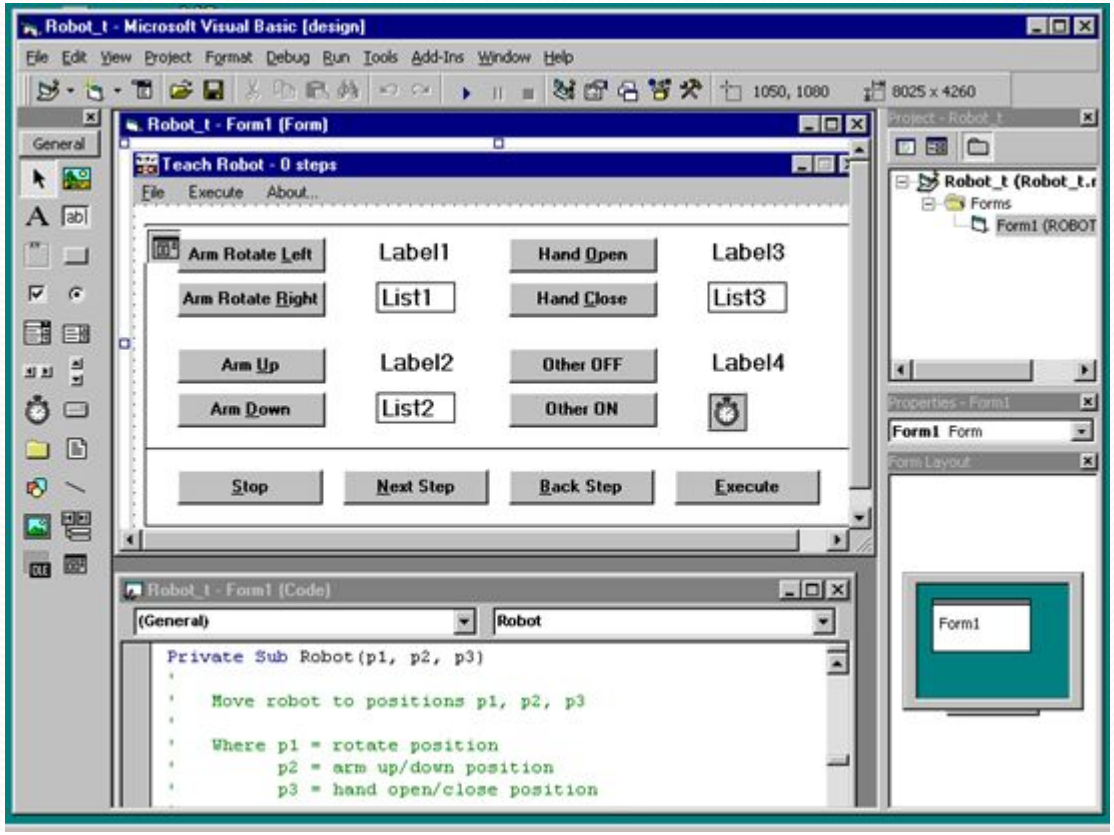


Будущие создатели Microsoft Пол Аллен (Allen, Paul; p. 1954) и Билл Гейтс (Gates, William; p. 1955) познакомились с Бэйсиком, работая в компьютерном классе школы в Сиэтле (снимок 1968 г.)

Начав с Бэйсика, компания Microsoft превратилась в крупнейшую софтверную империю, а Билл Гейтс –стал самым богатым человеком на планете

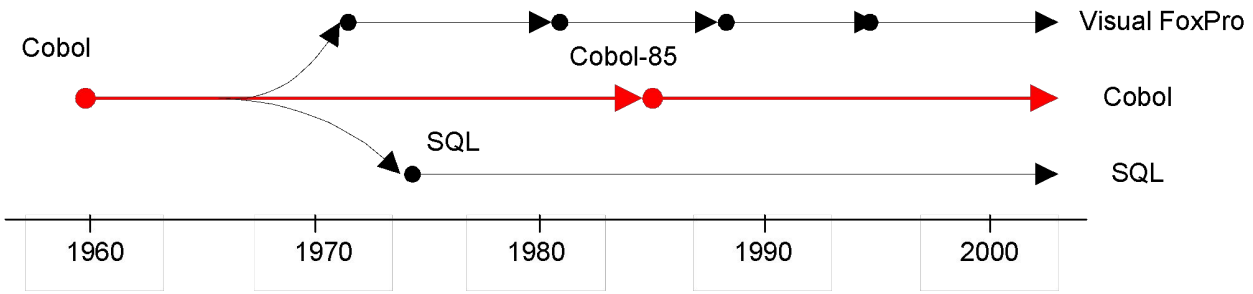
Штаб - квартира корпорации Microsoft в Редмонде (пригород Сиэтла)

Visual Basic от Microsoft



На протяжении нескольких десятилетий Visual Basic оставался фирменный языком компании Microsoft. В начале 1990-х годов он стал объектным и приобрел средства визуального проектирования

Cobol – язык для бухгалтеров



COBOL = COmmon Business-Oriented Language

На фото: разработчики языка Cobol у шуточного обелиска, присланного в их адрес в качестве намека на безнадежно медленную работу, способную похоронить саму идею. Справа внизу – Грейс Хоппер

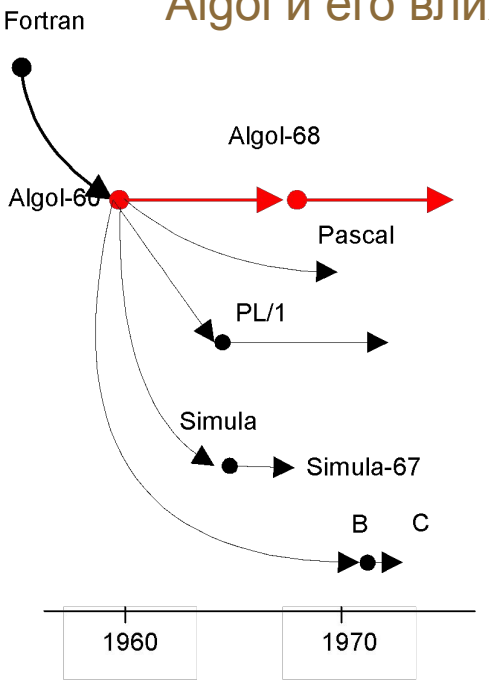
COBOL PROGRAMMING

Основные свойства языка Cobol:

- независимость программ от оборудования;
- независимость программ от данных;
- сложные структуры данных;
- синтаксис, приближенный к естественному английскому языку.

Языки и системы программирования

Algol и его влияние на языки программирования



ALGOL = ALGOritmic Language

В 1958 году в Цюрихе (Швейцария) состоялась международная конференция, предложившая проект нового универсального международного языка программирования Algol-58. В 1960 году на парижской конференции была принята окончательная версия под названием Algol-60.

На снимке: участники парижской конференции голосуют за Алгол-60.

Algol и его влияние на языки программирования

Основные свойства языка Algol-60:

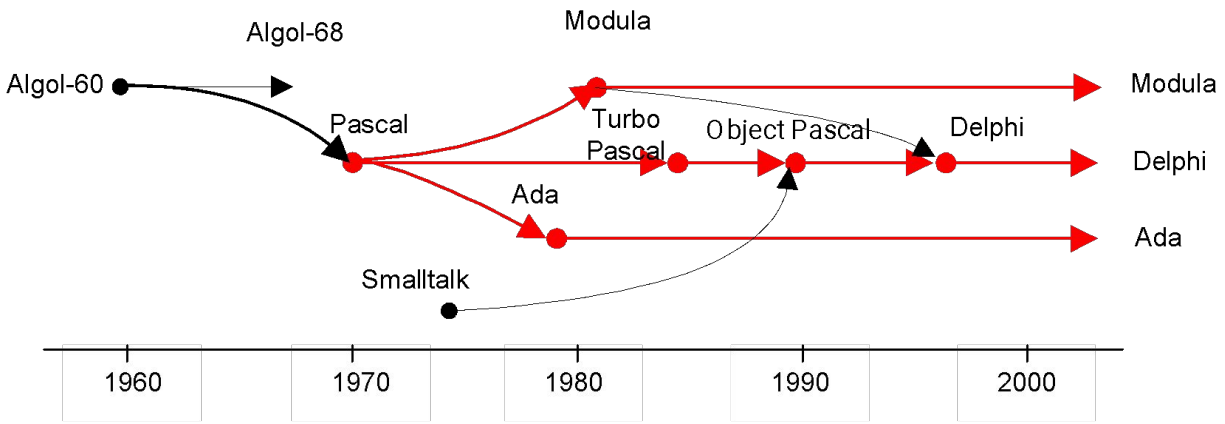
- машинная независимость;
- формальный синтаксис;
- описание переменных и блочная структура;
- рекурсия

Нормальная форма Бэкуса-Наура (БНФ)

<цифра>::= 1|2|3|4|5|6|7|8|9|0

<целое без знака>::= <цифра>| <цифра> <целое без знака>

Pascal и его потомки



Член комитета по Алголу-68 Никлаус Вирт (Wirth, Niklaus; p. 1934) был против принятия переусложненного стандарта.

В знак доказательства своей правоты он разработал в 1971 г. простой и ясный алголоподобный язык, предназначенный прежде всего для обучения студентов в Федеральном техническом университете в Швейцарии. В честь изобретателя первой вычислительной машины Вирт назвал язык Паскалем.

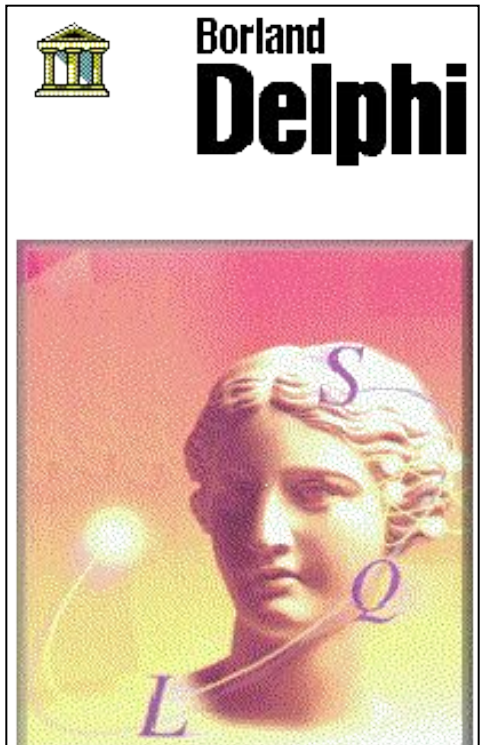
Паскаль (Pascal)

Никлаус Вирт (*Niklaus Wirth*) - швейцарский учёный, один из известнейших теоретиков в области разработки языков программирования, профессор компьютерных наук (ETH), Лауреат премии Тьюринга 1984 года. Разработал: Паскаль, Модула-2, Оберон.

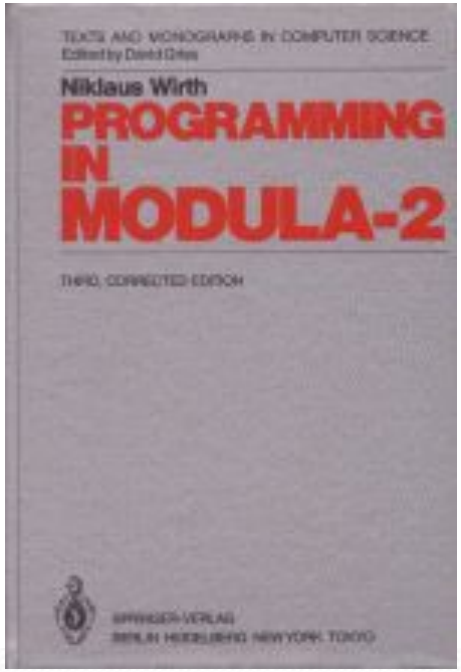
Turbo Pascal

Новую жизнь языку Pascal дал Филипп Кан (Kahn, Philippe; p. 1938) – создатель компилятора Turbo Pascal для IBM PC и основатель компании Borland (1984 г.)

Delphi - потомок Pascal



Среда разработки Delphi фирмы Borland объединила передовые достижения технологии программирования: объектное расширение языка Pascal, визуально-событийное проектирование, модульное структурирование и раздельная компиляция.



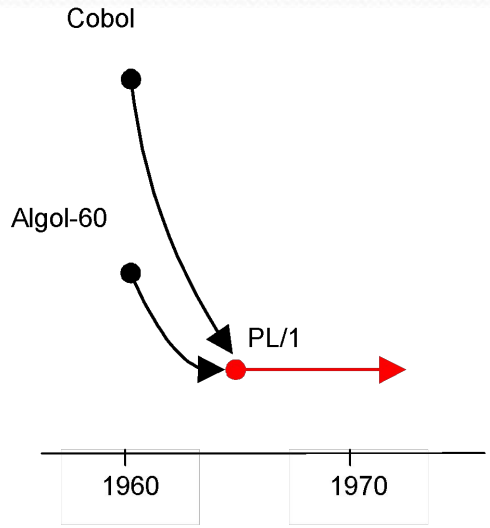
В отличие от учебного Паскаля, язык программирования Modula-2, предложенные Никлаусом Виртом, изначально предназначался для профессионального применения

Pascal и его потомки

В 1975 году Министерство обороны США приняло решение разработать стандартный язык для программирования сложных и ответственных военных приложений. Был объявлен широкий международный конкурс, в котором приняли участие 15 групп разработчиков. В результате нескольких туров в мае 1979 года выявился победитель — французская фирма C.I.I., руководитель проекта Жан Ихбиа (Ichbiah, Jean).

Снимок сделан на II конференции по истории языков программирования, 1993 г.

Суперязык PL/1 – самый сложный язык

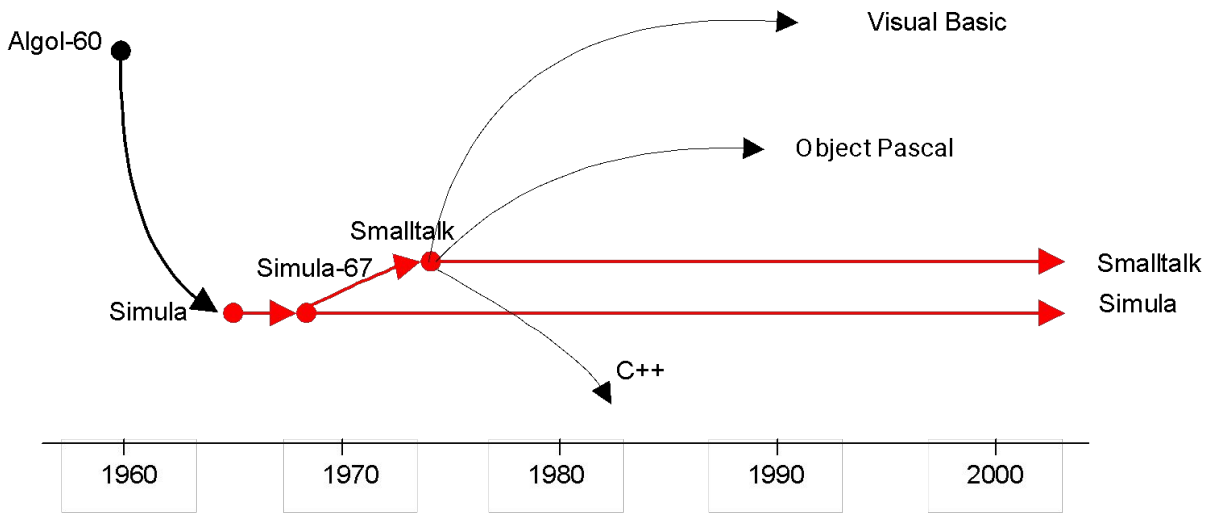


PL/1 = Programming Language One

Язык PL/1 был частью амбициозного проекта IBM S/360, он создавался в спешке и представлял собой механическую смесь идей из многих языков. Критики сравнивали его с елкой со множеством украшений.

```
EXAMPLE: PROCEDURE OPTIONS (MAIN);
ON ENDFILE (SYSIN) GO TO ENDING;
P1:     GET LIST (A, B, C);
        D = B*B — 4*A*C;
        E = —B/(A+A);
        IF D<0 THEN DO;
          X1, X2 = E;
          Y1 = SQRT(—D)/(A+A);
        END;
        ELSE DO;
          R = SQRT(D)/(A+A);
        …
        Y1 = 0;
        END;
        Y2 = —Y1;
        PUT LIST (X1, Y1, X2, Y2);
        GO TO P1;
ENDING:;
END EXAMPLE;
```

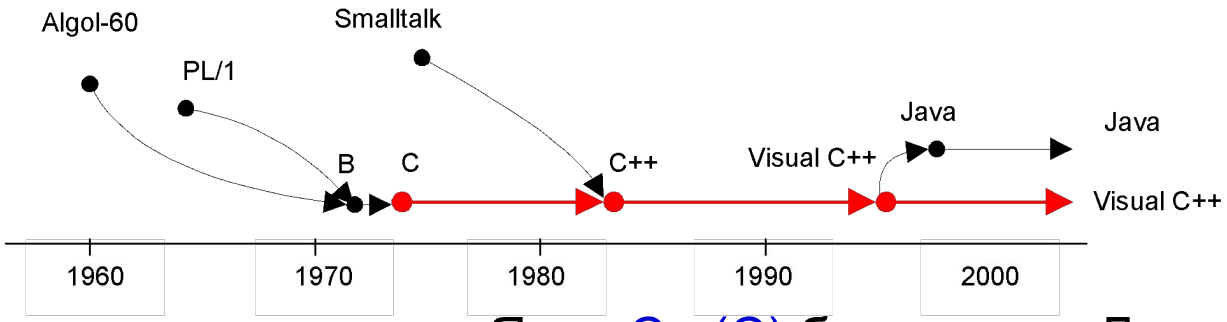
Simula и Smalltalk – революция в программировании – Объектно-Ориентированное программирование



Simula = SIMULAlation

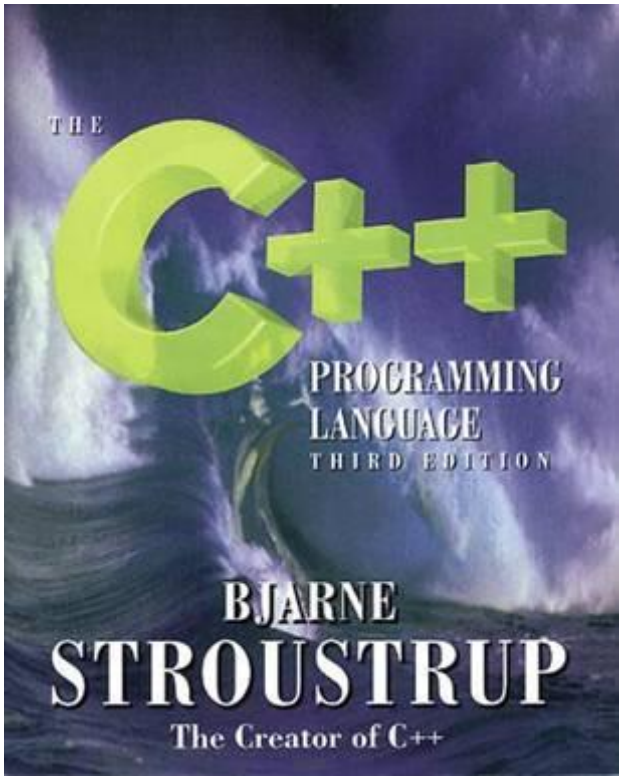
За разрабртку языка Simula Кристен Нигорд (Nygaard, Kristen; 1926-2002), на снимке слева, и Оле-Йохан Дал (Dahl, Ole-Johan; 1931-2002) были удостоены высшей награды компьютерного сообщества – медали Тьюринга

C – язык для профессионалов



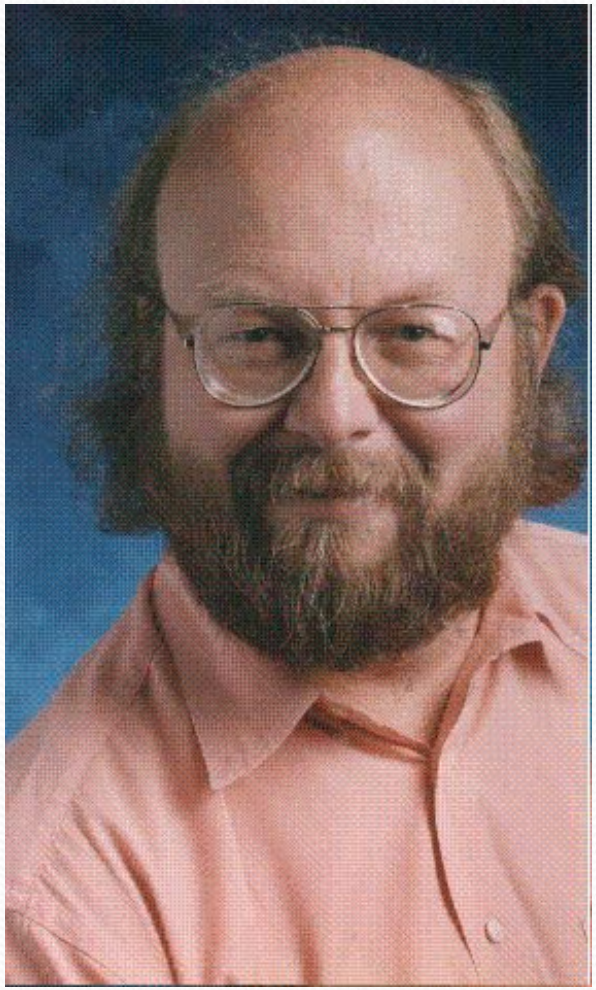
Язык Си (C) был создан Деннисом Ричи (Ritchie, Dennis M.; p. 1941) в 1973 году в Bell Labs в ходе разработки операционной системы UNIX. Он развивал язык Би (B), который основывался на созданном в Кембриджском университете языке BCPL (от Basic Combined Programming Language), который в свою очередь был потомком Алгола-60

C – язык для профессионалов



Бьярн Страуструп (Stroustrup, Bjarne; p. 1950) ввел в язык C объекты и превратил его в C++

Java – дитя интернета

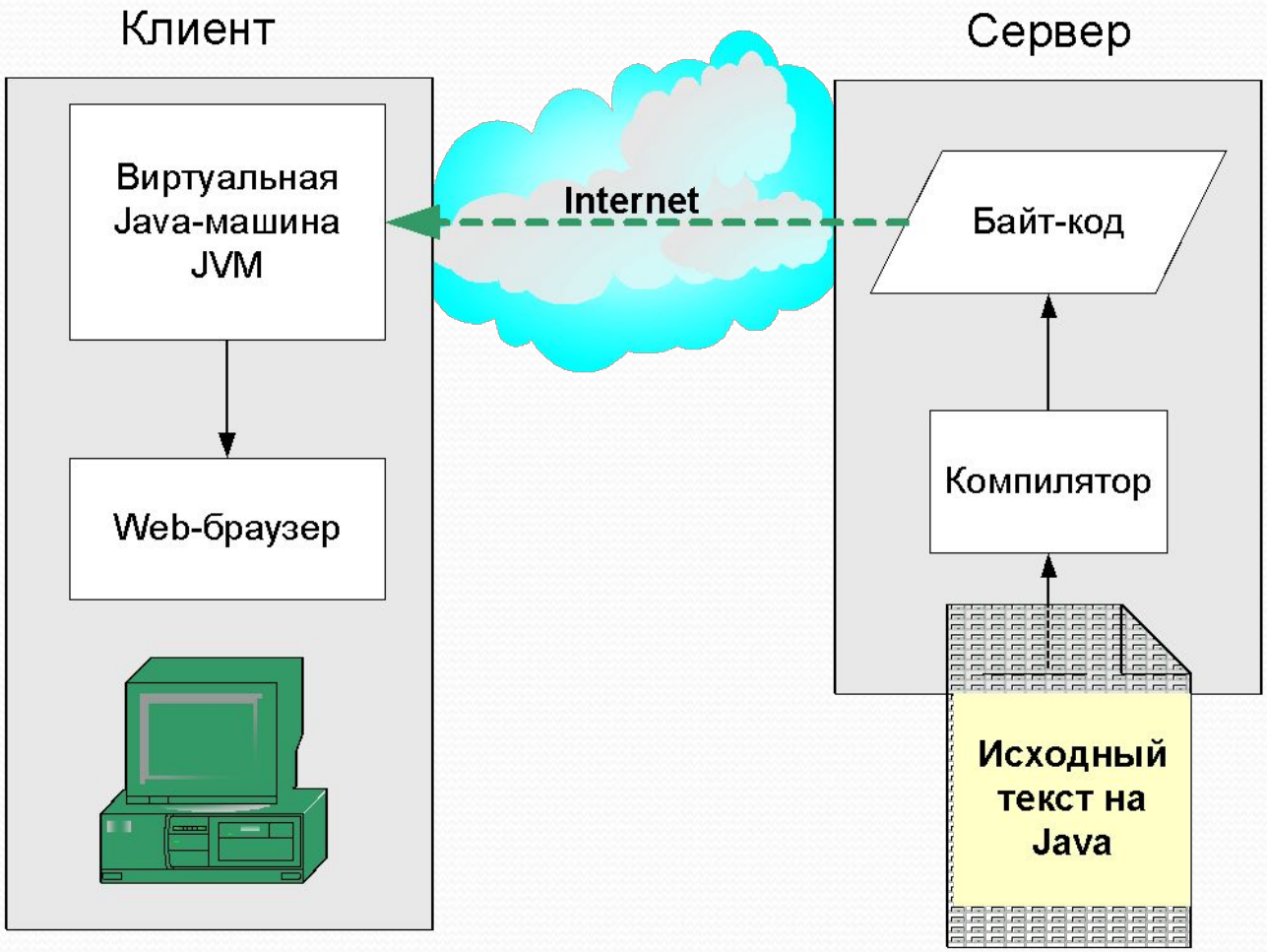


В 1995 г. фирма Sun Microsystems представила язык Java для программирования в интернете.

Он возник в ходе реализации проекта Oak («Дуб»), целью которого было создание системы программирования бытовых микропроцессорных устройств.

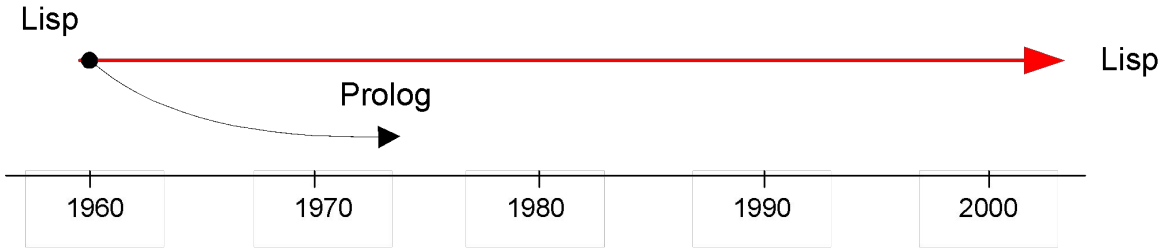
Джеймс Гослинг (Gosling, James) – автор Java.

Java-апплеты



Java - технология

Долгожитель Lisp – инструмент функционального программирования



Дж. Маккарти и А.П. Ершов
Снимок 1975 г.

Lisp = LISt Processing

Язык Lisp создан в 1960 году Джоном Маккарти (McCarthy, John; p. 1927) в Массачусетском технологическом институте на теоретическом фундаменте лямбда-исчисления, предложенного еще в 1930 году известным американским логиком Алонзо Черчем.

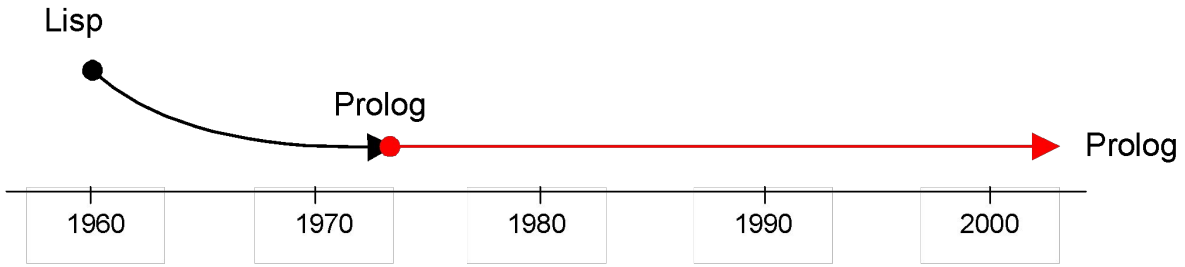
Scheme – 1975 год

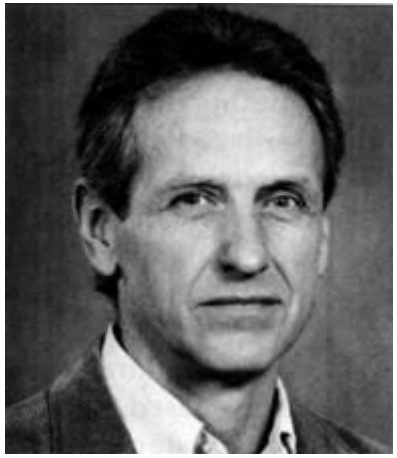
Функциональный язык программирования, один из двух наиболее популярных в наши дни диалектов языка Лисп (другой популярный диалект — это Common Lisp). Авторы языка Scheme — Гай Стил (англ. Guy L. Steele) и Джеральд Сассмен (англ. Gerald Jay Sussman) из Массачусетского технологического института — создали его в середине 1970-х годов.

Продолжения - *continuation*

состояние программы в определённый момент, которое может быть сохранено и использовано для перехода в это состояние. Продолжения содержат всю информацию, чтобы продолжить выполнения программы с определённой точки.

Prolog – несостоявшаяся мечта ЭВМ V поколения





Prolog = PROgramming for LOGic

Теоретические основы языка были разработаны Робертом Ковальским (Kowalski, Robert) в Эдинбургском университете (Шотландия) в конце 1960-х годов

Первая практическая реализация языка осуществлена Аленом Кольмари (Colmerauer, Alain) в Марсельском университете (Франция) в 1972 г.

Prolog – несостоявшаяся мечта ЭВМ V поколения

Факты:

муж (петя), муж (ваня),

муж (коля), жен (таня), жен (маша),

мать (ваня, таня), отец (ваня, петя),

отец (маша, ваня), отец (коля, ваня).

Правила вывода:

родитель (X, Y) :— отец (X, Y)

родитель (X, Y) :— мать (X, Y)

дед (X, Y) :— родитель (X, Z), отец (Z, Y)

брат (X, Y) :— муж (Y), родитель (X, Z),
родитель (Y, Z), X<>Y

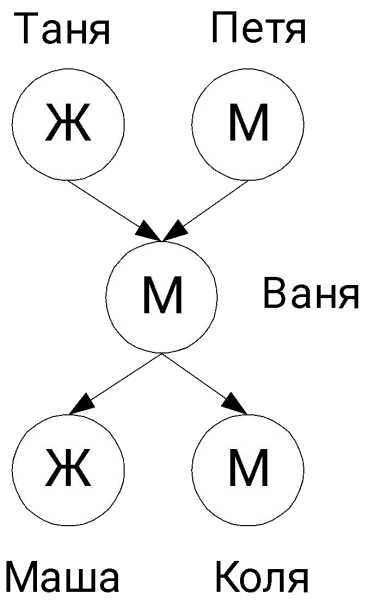
Примеры диалога:

GOAL> дед (коля, X) *Кто дед Коли?*

X = Петя

GOAL> брат (маша, X) *Кто брат Маши?*

X = Коля



Описание предметной области семейных отношений на языке Prolog

ЭВМ V поколения

Концептуальные отличия ЭВМ V поколения:

• новая технология производства микросхем, знаменующая переход от кремния к арсениду галлия, и дающая возможность на порядок повысить быстродействие основных логических элементов;

• новая архитектура (не фон-неймановская);

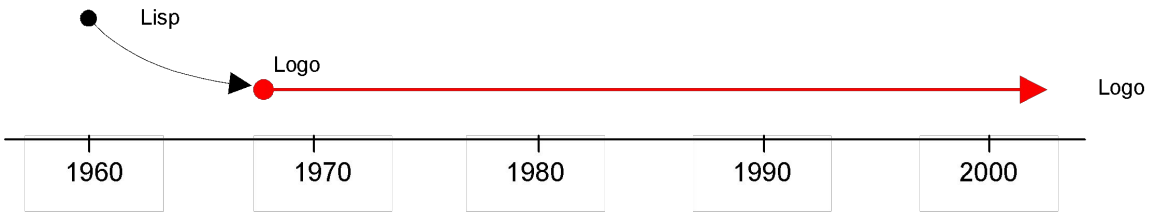
• новые способы ввода-вывода информации — распознавание и синтез речи и образов;

• отказ от традиционных алгоритмических языков программирования (Фортран, Алгол и т. п.) в пользу декларативных;

• ориентация на задачи искусственного интеллекта с автоматическим поиском решения на основе логического вывода.

Структура ЭВМ V поколения

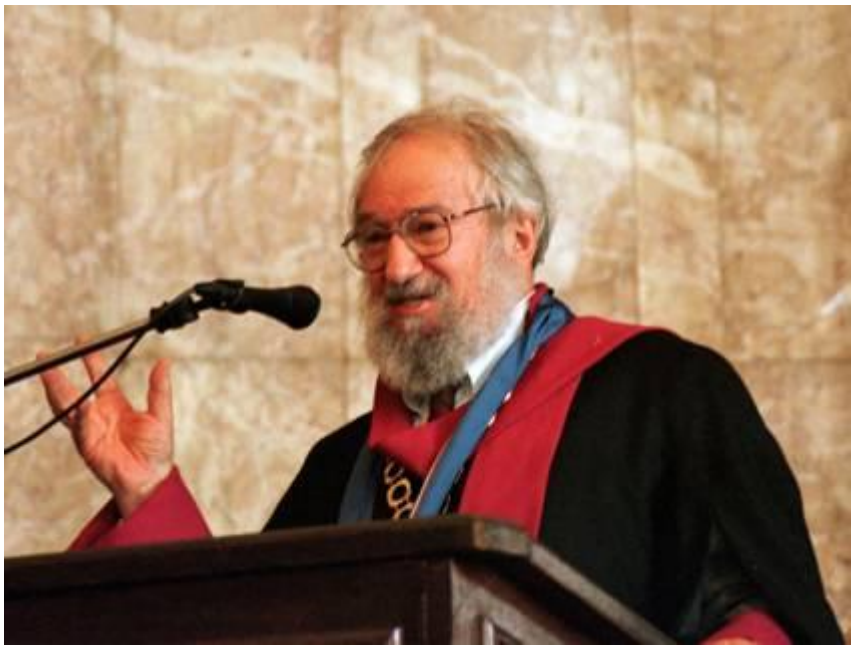
Logo – язык для самых маленьких



Язык Logo, изобретен в 1967 г. в MIT выдающимся математиком и педагогом Сеймуром Пейпертом (Papert, Seymour; p. 1928).

Пейперт в 1958-1963 годах работал в Женеве у знаменитого психолога Жана Пиаже (Piaget, Jean), где занимался детьми и природой их мышления.

Идейной основой Logo является язык Lisp



На фото: Сеймур Пейперт получает степень почетного доктора Софийского университета (1999 г.)

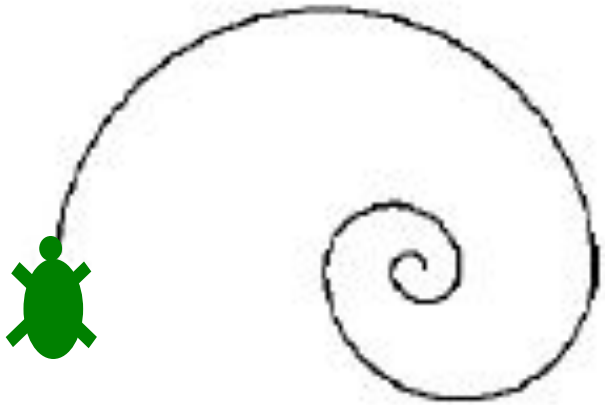
Языки и системы программирования

Logo – язык для самых маленьких

```
это дуга :шаг :число_шагов
   повтори :число_шагов
     [вперед :шаг направо 10]
Конец

это спираль :шаг
   если :шаг < 1 [стоп]
   дуга :шаг 18
   спираль :шаг / 2
конец
```

Цикл



Процедура с параметром

Программа на Logo управляет черепашкой, оставляющей видимый след. С помощью зрительных образов интерпретируются все базовые структуры программирования

Рекурсия

Парадигмы программирования

Основные парадигмы программирования:

- программирование в машинных кодах (Assembler);
- процедурное программирование (Fortran, Basic, Cobol, Algol, Pascal, Ada, C, Logo, FoxPro);
- объектно-ориентированное программирование (Simula, Smalltalk, Object Pascal, C++, Java, C#);
- визуально-событийное программирование (Visual Basic, Delphi, Visual C++, Visual Java, Visual FoxPro);
- функциональное программирование (Lisp);
- логическое программирование (Prolog).
- аспектно-ориентированное программирование.
- предметно-ориентированное программирование.
- субъектно-ориентированное программирование.

Python

Текст философии:

- Красивое лучше, чем уродливое.
- Явное лучше, чем неявное.
- Простое лучше, чем сложное.
- Сложное лучше, чем запутанное.
- Плоское лучше, чем вложенное.
- Разреженное лучше, чем плотное.
- Читаемость имеет значение.
- Особые случаи не настолько особые, чтобы нарушать правила.
- При этом практичность важнее безупречности.
- Ошибки никогда не должны замалчиваться.
- Если не замалчиваются явно.
- Встретив двусмысленность, отбрось искушение угадать.
- Должен существовать один — и, желательно, *только* один — очевидный способ сделать это.
- Хотя он поначалу может быть и не очевиден, если вы не голландец
- Сейчас лучше, чем никогда.
- Хотя никогда зачастую лучше, чем *прямо* сейчас.
- Если реализацию сложно объяснить — идея плоха.
- Если реализацию легко объяснить — идея, *возможно*, хороша.
- Пространства имён — отличная штука! Будем делать их побольше!

Влияние других языков на Python

Появившись сравнительно поздно, Python создавался под влиянием множества языков программирования:

- ABC (англ.) — отступы для группировки операторов, высокоуровневые структуры данных (map) (фактически, Python создавался как попытка исправить ошибки, допущенные при проектировании ABC);
- Modula-3 — пакеты, модули, использование `else` совместно с `try` и `except`, именованные аргументы функций (на это также повлиял Common Lisp);
- Си, C++ — некоторые синтаксические конструкции (как пишет сам Гвидо ван Россум — он использовал наиболее непротиворечивые конструкции из C, чтобы не вызвать неприязнь у Си-программистов к Python[11]);
- Smalltalk — объектно-ориентированное программирование;
- Lisp — отдельные черты функционального программирования (`lambda, map, reduce, filter` и другие);
- Fortran — срезы массивов, комплексная арифметика;
- Miranda — списочные выражения;
- Java — модули logging, unittest, threading (часть возможностей оригинального модуля не реализована), xml.sax стандартной библиотеки, совместное использование finally и except при обработке исключений, использование @ для декораторов;
- Icon — генераторы.

Большая часть других возможностей Python (например, байт-компиляция исходного кода) также была реализована ранее в других языках.