# Введение в языки программирования

## История языков программирования

### Предтечи.

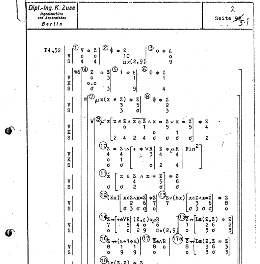
Пожалуй, первым языком программирования, как языке записи алгоритмов, для последующего выполнения на вычислительной машине, можно считать псевдомашинный алгоритмический код Ады Лавлейс, для разностной машины Беббиджа.

Ада Лавлейс (1815—1852), дочь известнейшего английского поэта лорда Байрона. Получила математической образование. Была хорошо знакома с учеными и выдающимися деятелями своего времени: Чарльзом Бэббиджем, Майклом Фарадеем, Дэвидом Брюстером, Чарльзом Уитстоном, Чарльзом Диккенсом и другими.

Один из алгоритмов, описанных ею — алгоритм вычисления чисел Бернулли, был признан как специально написанная программа для исполнения на вычислительной машине (разностной машине Беббиджа). Несмотря на то, что сама машина при жизни изобретателя и Ады не была построена, эта программа признана первой в мире, а сама Ада Лавлейс считается первым программистом.

В своих записях она рассуждала о том, что, подобно тому, как Жаккардов ткацкий станок может ткать цветы и листья, аналитическая машина способна создавать алгебраические формулы, а в перспективе — писать музыку, писать картины — и укажет «науке такие пути, какие нам и не снились».

В 40 годы, в Германии, инженер Конрад Цузе спроектировал и построил вычислительную машину Z4, параллельно с созданием машины, он разрабатывал первый в мире высокоуровневый язык программирования, созданный. В переводе на русский это название соответствует выражению «планируемые вычисления» или «запланированные вычисления», то есть «программирование».

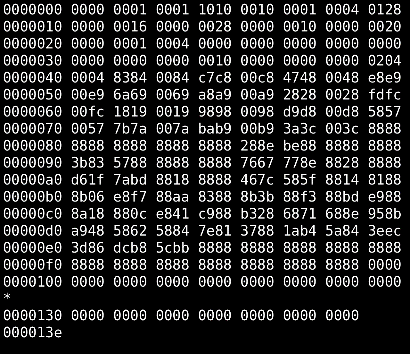
Планкалкюль поддерживал операции назначения, вызов подпрограмм, условные операторы, итерационные циклы, арифметику с плавающей запятой, массивы, иерархические структуры данных, утверждения, обработку исключений и многие другие вполне современные средства языков программирования.

### Машинный язык

Электронно-вычислительная машина, ЭВМ, компьютер — все это одно название современных вычислительных устройств. Физические принципы работы основаны на хранении и передачи электрических напряжений, а значит имеющих возможность передать только один бит информации.

1 бит информации — символ или сигнал, который может принимать два значения: включено или выключено, да или нет, высокий или низкий, заряженный или незаряженный; в двоичной системе исчисления это 1 (единица) или 0 (ноль). Это минимальное количество информации, которое необходимо для ликвидации минимальной неопределенности.

Этот набор битов информации, то есть набор нулей и единиц и есть машинный язык — единственный который «понимает» компьютер на базовом уровне.

Первые годы развития ЭВМ, человеку было необходимо составлять программы на языке, понятном компьютеру, то есть в машинных кодах. Каждая команда состояла

* из кода операций
* адресов операндов,

выраженных в виде различных сочетаний единиц и нулей.

Итак, любая программа для процессора выглядела на то время как последовательность единиц и нулей.

На практике, для постоянного общения с компьютером, такой язык громоздок и неудобен. При пользовании им легко допустить ошибку, записав не в той последовательности 1 или 0. Программу очень трудно контролировать. Кроме того, при программировании в машинных кодах надо хорошо знать внутреннюю структуру ЭВМ, принцип работы каждого блока. И самое плохое в таком языке, что программы на данном языке — очень длинные последовательности единиц и нулей являются машинно-зависимыми, то есть для каждой ЭВМ необходимо было составлять свою программу, а также программирование в машинных кодах требует от программиста много времени, труда, повышенного внимания.

### Ассемблер

Уже в начале 50-х годов были разработаны мнемонические языки для описания машинных команд более-менее удобными для восприятия словесными сокращениями. Например, вместо набора нулей и единиц 00101100, которые могли обозначать сложение двух регистров процессора, появилась возможность написать add.

Ассемблер — язык программирования низкого уровня. Язык программирования низкого уровня — язык программирования, который ориентирован на конкретный тип процессора и учитывает его особенности. В данном случае «низкий уровень» не значит «плохой». Имеется в виду, что операторы языка близки к машинному коду и ориентированы на конкретные команды процессора. Появление языка ассемблера значительно облегчило жизнь программистов, так как теперь вместо рябящих в глазах нулей и единиц, они могли писать программу командами, состоящими из символов приближенных к обычному языку. Для того времени этот язык был новшеством и пользовался популярностью так как позволял писать программы небольшого размера, что при тех машинах критерий значительный.

Программа Hello World на assembler Intel:

; Hello World for Intel Assembler (MSDOS)

mov ax,cs

mov ds,ax

mov ah,9

mov dx, offset Hello

int 21h

xor ax,ax

int 21h

Hello:

db "Hello World!",13,10,"$"

Вместе с тем, отметим, что нет как такого одного Ассемблера. Дело в том, что ассемблер это по сути «очеловеченный» машинный язык, а значит, что для каждого типа процессора существует свой Ассемблер. Конечно, у них есть общие команды и принципы, но все-таки есть и различия.

Также надо понимать, что программирование на языке ассемблера, хоть и стало понятнее, однако, по-прежнему оставалось очень трудоемким, ведь, по сути, приходилось вручную описывать каждый шаг процессора. Все это привело к созданию языков третьего поколения — языков высокого уровня.

Программа Hello World на assembler ARM:

/\* Hello world in ARM assembly (Android devices) \*/

.data

msg:

.ascii "Hello, World!\n"

len = . - msg

.text

.globl \_start

\_start:

mov %r0, $1

ldr %r1, =msg

ldr %r2, =len

mov %r7, $4

swi $0

mov %r0, $0

mov %r7, $1

swi $0

Но сегодня применение ассемблера не прекратилось. Он используется в написании отдельных фрагментов программ или иногда в написании самих программ. Примеров может быть много, но самые яркие это использование ассемблера в написании драйверов, игровых движков и загрузчиков ОС (операционных систем).

Также ассемблером пользуются хаккеры, для написания вирусов, шпионских программ (троянов), различных «отмычек». Это связанно с тем, что объем кода таких программ гораздо меньше, а значит и скорость работы значительно выше скорости программы, написанной на языке программирования высокого уровня.

Программа Hello World на assembler Win32:

; Hello world in Assembler for the Win32 architecture

TITLE Hello world in win32. Tasm

VERSION T310

Model use32 Flat,StdCall

start\_code segment byte public 'code' use32

begin:

Call MessageBox, 0, offset sHallo, offset caption, 0

Call ExitProcess, 0

start\_code Ends

start\_data segment byte public 'data' use32

sHallo db 'Hello world',0

caption db "Hi",0

start\_data Ends

End begin

По этой же причине (малый объем и быстродействие) и разработчики антивирусов так же используют ассемблер в некоторых модулях своих программ.

Важно заметить, что термин Ассемблер относится не только к языкам программирования, но и к специальным программам, которые переводят программу с языка Ассемблера и создают — «собирают» (assembly) исполняемую программу на машинном языке.

### Языки третьего поколения (языки высокого уровня)

В 50-60 годы возникло понимание того, что привязка языка программирования к определенной архитектуре процессора (иными словами — к одной платформе), тормозит развитие программирования. Стали разрабатываться и внедряться языки высокого уровня, которые уже были кроссплатформенными в плане синтаксиса. То есть текст программы на таком языке одинаково подходил для любой платформы. Другое дело, что такой текст должен был быть «переведен» — оттранслирован — в машинный код конкретной архитектуры процессора и операционной системы. Это осуществляли специальные программы трансляторы (компиляторы и интерпретаторы).

#### Фортран

Первым языком высокого уровня признан язык Фортран (FORTRAN — **Form**ula **Tran**slation). Созданный в 1954-57 годах специалистами компании IBM под руководством Джона он стал практически сразу же популярен среди ученых и инженеров для написания программ с интенсивными вычислениями. Включение типа данных «комплексное число», сделало его особенно удобным для технических приложений. В Советском Союзе в конце 60-х готов был написан собственный компилятор для языка Фортран.

Программа Hello World на Fortran:

C Hello World in Fortran

PROGRAM HELLO

WRITE (\*,100)

STOP

100 FORMAT (' Hello World! ' /)

END

Фортран широко используется и сегодня. В основном для научных и инженерных вычислений. Он прекрасно подходит для решения численных задач, так как за время его существования было написано множество библиотек. Среди учёных ходит такая присказка, что любая математическая задача уже имеет решение на Фортране, и, что недалеко от истины: среди тысяч фортрановских пакетов и библиотек, можно найти и пакеты для работы с матрицами, и пакет для решения сложных интегральных уравнений, и многие другие.

Программа Hello World на Fortran90:

! Hello World in Fortran 90 and 95

PROGRAM HelloWorld

WRITE(\*,\*) "Hello World!"

END PROGRAM

#### Алгол

Фортран создавался в США, и в Европе небезосновательно считали, что IBM начнет в качестве монополиста диктовать условия. В 1958 году был создан первый стандарт языка Algol (algorithmic language). Вскоре его доусовершенствовали, в результате появился язык Алгол-60.

В Алголе появилось представление о программе не как о свободной последовательности команд, а как о блочной структуре, состоящей из чётко описанных и отделённых друг от друга частей. Основной блок программы на Алголе — это сама главная программа. Она содержит свою исполняемую часть, заключённую в главный блок, а также описания подпрограмм. Каждая подпрограмма — это программа в миниатюре, имеющая собственные, описанные внутри неё данные, однозначно определённый интерфейс в виде имени и списка формальных параметров, и блок кода. При этом в блоке могут выделяться подблоки.

Программа Hello World на Algol60:

'BEGIN'

'COMMENT' Hello World in Algol 60;

OUTPUT(4,'(''('Hello World!')',/')')

'END'

Были выделены структурные управляющие конструкции: ветвления, циклы, последовательные участки, исполняющие условно или многократно вложенные наборы операторов. То, что сегодня кажется очевидным, в те годы было настоящим прорывом.

#### LISP

LISP является аббревиатурой выражения LISt Processing — обработка списков. Идея языка была предложена Дж. Маккарти в 1960 году, и сводилось к тому, что обработка была направлена не на вычисление, а на обработку информации, оформленную в виде списков.

Программа Hello World на Lisp:

;;; Hello World in Emacs Lisp.

(defun hello-world()

"Display the string hello world."

(interactive)

(message "hello world"))

В виде списков удобно представлять алгебраические выражения, графы, элементы конечных групп, множества, правила вывода и многие другие сложные объекты. Списки являются наиболее гибкой формой представления информации в памяти компьютеров. Неудивительно, что удобный язык Лисп, специально предназначенный для обработки списков, быстро завоевал популярность.

#### COBOL

COBOL (**co**mmon **b**usiness-**o**riented **l**anguage) — был разработан в 1959 году и предназначался прежде всего для написания программ для разработки бизнес приложений, а также для работы в экономической сфере.

Программа Hello World на Lisp:

\* Hello World in COBOL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

IDENTIFICATION DIVISION.

PROGRAM-ID. HELLO.

ENVIRONMENT DIVISION.

DATA DIVISION.

PROCEDURE DIVISION.

MAIN SECTION.

DISPLAY "Hello World!"

STOP RUN.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Создатели языка ставили своей целью сделать его машинонезависимым и максимально приближенным к естественному английскому языку. Обе цели были успешно достигнуты; программы на COBOL считаются понятными даже неспециалистам, поскольку тексты на этом языке программирования не нуждаются в каких-либо специальных комментариях (самодокументирующиеся программы).

#### BASIC

Бейсик был придуман в 1964 году преподавателями Дартмутского Колледжа Джоном Кемени и Томасом Курцем, и под их руководством был реализован командой студентов колледжа. Со временем, когда стали появляться другие диалекты, изначальную реализацию стали называть Dartmouth BASIC.

Программа Hello World на Basic:

10 REM Hello World in BASIC

20 PRINT "Hello World!"

Бейсик был спроектирован так, чтобы студенты могли без затруднений писать программы, используя терминалы с разделением времени. Он предназначался для более «простых» пользователей, не столько заинтересованных в скорости исполнения программ, сколько просто в возможности использовать компьютер для решения своих задач, не имея специальной подготовки.

Про Бейсик можно было бы написать трактат, как из малоизвестного интерпретатора он поднялся до невероятных высот популярности и после стал «притчей во языцех» как несовершенный и неудобный язык. Однако справедливости ради отметим, что это была прекрасная история, а главное современное перерождение в Visual Basic подарило ему вторую жизнь (хотя конечно новая версия практически не имеет ничего общего с классическим Basic)