



Ростовский Государственный Строительный Университет

ЯЗЫКИ

Программирования Высокого Уровня

Презентацию
подготовил

Колыванов Андрей
ученик 9 Б класса

Введение

Создатели языка программирования формулируют понятие языка программирования. Чем более распространенным утверждением признаваемым большинством разработчиков, относится следующие:

представляет собой некоторый алгоритм в форме, понятной **Функция:** язык программирования предназначен для написания компьютерных программ, которые применяются для передачи компьютеру инструкций по выполнению того или иного вычислительного процесса и организации управления отдельными устройствами. синтаксических и семантических правил, используемых при

Задача языка программирования отличается от естественных языков тем, что предназначен для передачи команд и данных от человека компьютеру, в то время, как естественные языки используются для общения людей между собой. В принципе, можно обобщить определение «языков программирования» — это способ передачи данных, а также какие именно действия следует выполнять над команд, приказов, четкого руководства к действию, тогда как человеческие языки служат также для обмена информацией в различных обстоятельствах.

Исполнение: язык программирования может использовать специальные конструкции для определения и манипулирования структурами данных и управления процессом вычислений.

Введение

Список языков программирования: Классы языков программирования:

1. Неклассифицированные языки
1. Функциональные
2. XML-подобные языки программирования
2. Императивные
3. Структурные языки программирования
3. Стековые
4. Процедурные языки программирования
5. Веб-языки программирования
6. Аспектно-ориентированные языки
7. Декларативные
8. Объектно-ориентированные языки программирования
8. Динамические
9. Учебные. Описания интерфейсов
9. Языки программирования для промышленной автоматизации
10. Прототипные
10. Эзотерические языки программирования
11. Объектно-ориентированные
11. Стековые языки
12. Рефлексивные
12. Параллельные языки программирования
13. Логического программирования
14. Параллельного программирования
15. Сценарные (скриптовые)
16. Эзотерические

Введение

Неклассифицированные языки:

ABAP/4	JOVIAL	Робик
Awk	Jython	Рапира
BCPL	Mercury	УА (Упрощённый Алгол)
FoxPro	Linda	Nemerle
Tcl/Tk	Occam	ДРАКОН
ML	ПЛ/1	ORACLE
MQL4	PL/M	
PostScript	Pixilang	
PL/SQL	Планкалкюль	
Clarion	Scala	
Clean	xBase	
Clipper	Progress 4gl	
Curry	X++	
Gentee	Sieve	
GPSS	PureBasic — компилируемая модификация Basic	
DataFlex	Ассемблеры	
Erlang	Visual DataFlex	

Введение

XML-подобные языки программирования

ApplicationXML

Структурные языки программирования

Алгол

Алгол 68

Basic

QBASIC

Фортран

REXX

sh

Фокал

Введение

Процедурные языки программирования

Алгоритмический язык

Би (язык программирования)

Си

КОБОЛ

Limbo

Lua

Maple

MATLAB

Модула-2

Паскаль

Логические языки программирования

Prolog

Введение

Функциональные языки программирования

Лисп

Cat (Stack-oriented programming language)

Лого

Dylan

Haskell

OCaml

Scheme

РЕФАЛ

АПЛ

J

Норе

Стековые языки

PostScript

Forth

Параллельные языки программирования

МС#

Языки программирования для промышленной автоматизации

(стандарта IEC61131-3)

FBD

IL

ST или SCL

Sequential Function Chart

Ladder Diagram

SPCLK

Эзотерические языки программирования

Byter

Brainfuck

Befunge

INTERCAL

FALSE

Whitespace

Piet

Стандартизация языков программирования

Язык программирования может быть представлен в виде набора спецификаций, определяющих его синтаксис и семантику.

Для многих широко распространённых языков программирования созданы международные стандарты. Специальные организации проводят регулярное обновление и публикацию спецификаций и формальных определений соответствующего языка. В рамках таких комитетов продолжается разработка и модернизация языков программирования и решаются вопросы о расширении или поддержке уже существующих и новых языковых конструкций.

Типы данных

Во внутреннем представлении обычно данные в современных цифровых компьютерах сохраняются в бинарном виде (в двоичном виде). Данные, которые представляют информацию из реального мира (имена, банковские счета, измерения и др.) высокоуровневые концепции.

Особая система, по которой данные организуются в программе, эта система типов языка программирования; разработка и изучение систем типов известна под названием теория типов. Языки могут быть классифицированы как системы со статической типизацией и языки с динамической типизацией.

Статически-типизированные языки могут быть в дальнейшем подразделены на языки с обязательной декларацией, где каждая переменная и объявление функции имеет обязательное объявление типа, и языки с выводимыми типами. Иногда динамически-типизированные языки называются латентно типизированными

Структуры данных

Системы типов в языках высокого уровня позволяют определять сложные, составные типы, так называемые структуры данных. Как правило, структурные типы данных образуются как декартово произведение базовых (атомарных) типов и ранее определённых составных типов.

Основные структуры данных (списки, очереди, хэш-таблицы, двоичные деревья и пары) часто представлены особыми синтаксическими конструкциями в языках высокого уровня. Такие данные структурируются автоматически.

Семантика языков программирования

Существует несколько подходов к определению семантики языков программирования.

Наиболее широко распространены разновидности следующих: операционного (или так называемого математического), и деривационного (или аксиоматического).

При описании семантики в рамках операционного подхода обычно исполнение конструкций языка программирования интерпретируется с помощью некоторой воображаемой (абстрактной) ЭВМ.

Деривационная семантика описывает последствия выполнения конструкций языка с помощью языка логики и задания пред- и постусловий. Денотационная семантика оперирует понятиями, типичными для математики - множества, соответствия и др.

Парадигмы программирования

Язык программирования строится в соответствии с той или иной базовой моделью вычислений.

Несмотря на то, что большинство языков ориентировано на так называемую императивную модель вычислений, задаваемую так называемой фон-неймановской архитектурой ЭВМ, существуют и другие подходы. Прежде всего следует упомянуть языки со стековой вычислительной моделью (Forth, Factor, Postscript и др.), а также функциональное (Лисп, Haskell, ML и др.) и логическое программирование

В настоящее время также активно развиваются проблемно-ориентированные, декларативные и визуальные языки программирования.

Компилируемые и интерпретируемые языки

Классификация языков программирования по способу трансляции в машинный код. Языки программирования делятся на компилируемые и интерпретируемые. В компилируемых языках программа сначала переводится в машинный код, а затем выполняется. В интерпретируемых языках программа выполняется непосредственно с помощью интерпретатора.

В компилируемых языках программа сначала переводится в машинный код, а затем выполняется. В интерпретируемых языках программа выполняется непосредственно с помощью интерпретатора. Компилируемые языки позволяют создавать более эффективные программы, так как компилятор может оптимизировать код. Интерпретируемые языки позволяют создавать программы, которые выполняются быстрее, так как интерпретатор выполняет код по мере его выполнения.

Если программа написана на интерпретируемом языке, то интерпретатор выполняет ее текстовый код непосредственно, без предварительного перевода. При этом программа остается на исходном языке и не может быть запущена без интерпретатора. Можно сказать, что время исполнения программы код может так же динамически компилироваться во время исполнения. Однако, компилируемые, кроме того, они не могут выполняться без дополнительной программы-интерпретатора.

ЯЗЫКИ ВЫСОКОГО УРОВНЯ

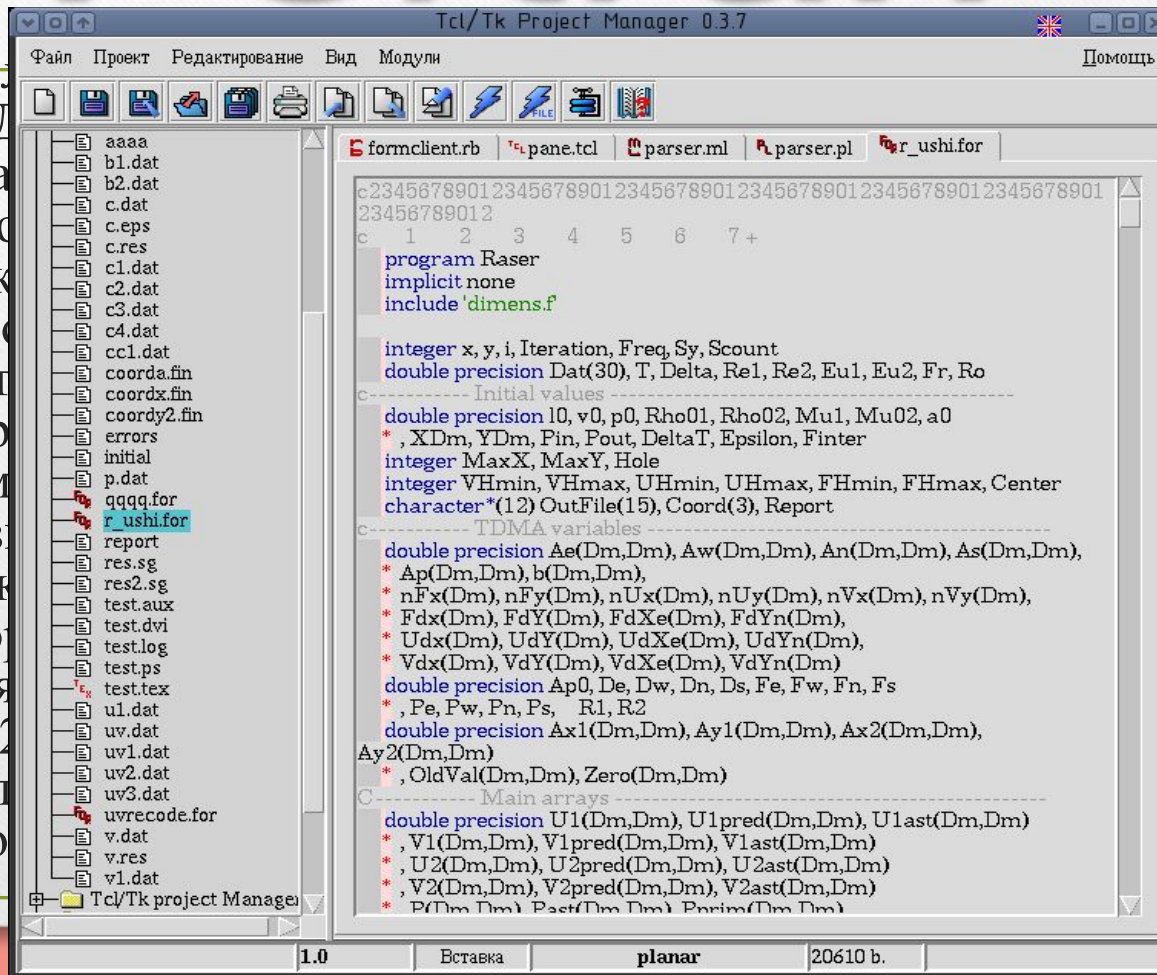
Высокоуровневый язык программирования - язык программирования, разработанный для быстроты и удобства использования программистом. Основная черта высокоуровневых языков - это абстракция, то есть введение смысловых конструкций, кратко описывающих такие структуры данных и операции над ними, описания которых на машинном коде (или другом низкоуровневом языке программирования) очень длинны и сложны для понимания.

Так, высокоуровневые языки стремятся не только облегчить решение сложных программных задач, но и упростить сортирование программного обеспечения. Использование разнообразных трансляторов и интерпретаторов обеспечивает связь программ, написанных при помощи языков высокого уровня, с различными операционными системами и оборудованием, в то время как их исходный код остаётся, в идеале, неизменным.

Язык высокого уровня

Fortran

Первый реал-тайм созданный для разработки серьезного программного языка высокого компилятор. Хотя в Фортране программирование жертву возмездия для этого языка от статических поэтому Фортраном организация Фортрана Fortran версия Фортрана компьютеро



ня
пражали
ительного
работке
хода.
тий
но в
. Однако
начиная
никами,
их
том
ная
ых супер

Cobol

Apply Filters

Click Next to analyze and convert the source files. You may also char

Property	Value
<input checked="" type="checkbox"/> cobolFilter	
<input type="checkbox"/> compilerSourceSettings	
COMP_6 switch (for MICROFOCUS compiler ...	COMP-6'2'
Compiler source	Default / N
Storage mode (for MICROFOCUS compiler s...	NOIBMCOM
Ignore after column 72	true
Ignore first 6 columns	false
Prefix nested columns	true
Replace hyphens (-) in record and field names ...	true
<input type="checkbox"/> template	
Case sensitive	false
Find	
Replace with	

New Wizard - Cobol Copybook

Steps

1. Select Wizard Type
2. **Select Cobol Copybook Files**
3. Configure Converter Options

Select Cobol Copybook Files

Browse Files

Look In:

- 510 Sample Projects
- 510 Samples Packages
- CoCoCo
- Doc Plans and Outlines
- esr
- Help - eWays
- HTTPS
- Informix 5.1 config chapter
- jar files
- JDBC
- My Music
- My Pictures
- My Shapes
- Norton 10.01
- RFID
- SeeBeyond Forms
- Shared
- SNA

File Name:

Files of Type: Cobol Copybook Files (.cpy, .cob, .cbl, .ccc)

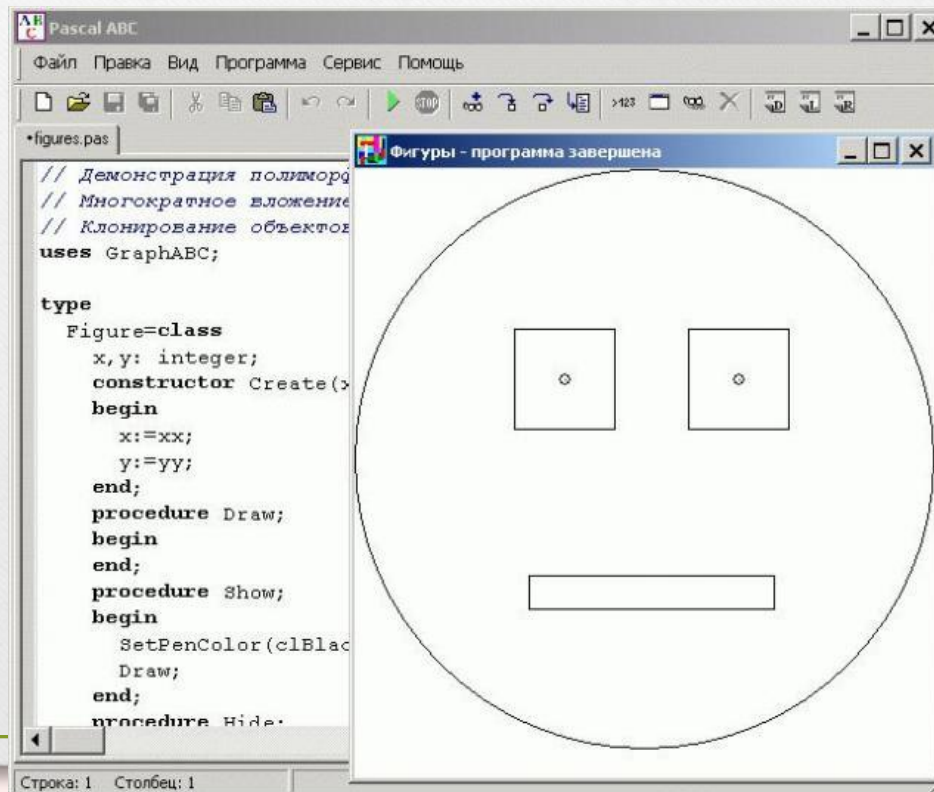
Select Files

Алгол

Алгол (англ. Algol от англ. algorithmic - алгоритмический и англ. language - язык) - название ряда языков программирования, применяемых при составлении программ для решения научно-технических задач на ЭВМ. Разработан комитетом по языку высокого уровня IFIP в 1958-1960 гг. (Алгол-58, Алгол-60); усовершенствован в 1964-1968 гг. (Алгол 68). Алгол относится к языкам высокого уровня и позволяет легко переводить алгебраические формулы в программные команды. Алгол был популярен в Европе, в том числе в СССР, в то время как сравнимый с ним язык Фортран был распространен в США и Канаде. Оказал заметное влияние на все разработанные позднее императивные языки программирования - в частности, на язык Pascal.

Pascal

Один из наиболее известных языков программирования, широко применяется в промышленном программировании, обучении программированию в высшей школе, является базой для большого числа других языков. Был создан Никлаусом Виртом в 1970, после его участия в работе комитета разработки стандарта языка Алгол-68.



```

Pascal ABC
Файл Правка Вид Программа Сервис Помощь
*figures.pas
// Демонстрация полиморфизма
// Многократное вложение
// Клонирование объектов
uses GraphABC;

type
  Figure=class
    x,y: integer;
    constructor Create(x,y: integer);
  begin
    x:=xx;
    y:=yy;
  end;
  procedure Draw;
  begin
  end;
  procedure Show;
  begin
    SetPenColor(clBlack);
    Draw;
  end;
  procedure Hide;
  end;
end;

```

Фигуры - программа завершена

Pascal

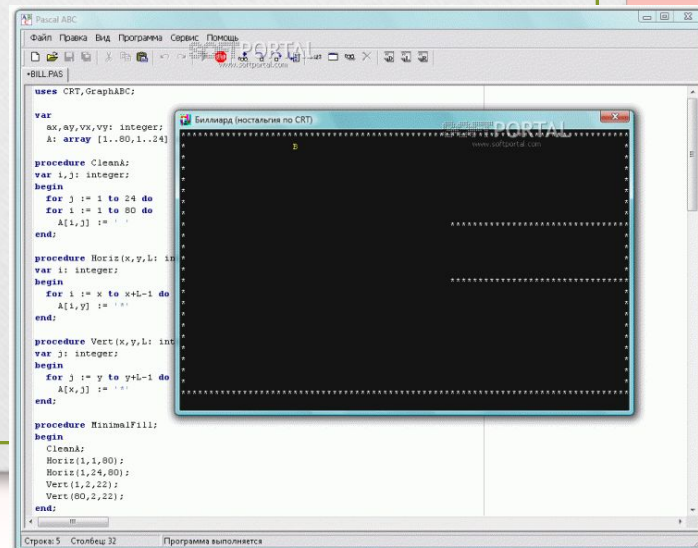
Особенностями языка являются строгая типизация и наличие средств структурного (процедурного) программирования. Паскаль был одним из первых таких языков. По мнению Н. Вирта, язык должен способствовать дисциплинированию

программирования, поэтому, наряду со строгой типизацией, в Паскале сведены к минимуму возможности манипуляций с памятью, а язык синтаксис автор постарался сделать настолько простым, чтобы даже начинающий знакомство с языком не должен). Название языку дано в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля. Один из первых языков, для которых была создана реализация «на самом себе» — компилятор Паскаля был написан на самом Паскале. В начале 1970-х годов для переноса Паскаль-систем на различные аппаратные платформы была создана система Pascal-P, в которой был единый компилятор Паскаля в промежуточный язык (P-код) и для каждой платформы создавался быстрый интерпретатор P-кода. Заимствование этой системы привело к созданию системы UCSD Pascal в Университете Сан-Диего (Калифорния, США), намного позже её идеи были заимствованы создателями языка Java (байт-код, компиляция в байт-код, интерпретатор байт-кода).

Pascal

Тем не менее, первоначально язык имел ряд ограничений: невозможность передачи функциям массивов переменной длины, отсутствие нормальных средств работы с динамической памятью, ограниченная библиотека ввода-вывода, отсутствие средств для подключения функций написанных на других языках, отсутствие средств отдельной компиляции и т. п. Наиболее бросающийся в глаза недостаток синтаксиса — некритически заимствованная из Алгола структура управляющих конструкций (операторов `if` и циклов), требующая, как правило, постоянного использования составных операторов «`begin — end`». Полный разбор недостатков языка Паскаль был выполнен Брайаном Керниганом в статье «Почему Паскаль не является моим любимым языком программирования» (интересно, что эта статья вышла в начале 1980-х, когда уже существовал язык Модула-2, потомок Паскаля, избавленный от большинства его пороков, а также более развитые диалекты Паскаля). Некоторые недостатки Паскаля были исправлены в ISO-стандарте 1982 года, в частности, в языке появились открытые массивы, давшие возможность использовать одни и те же процедуры для обработки одномерных массивов различных размеров.

Необходимо заметить, что многие недостатки языка не проявляются или даже становятся достоинствами при обучении программированию. Кроме того, по сравнению с основным языком программирования в академической среде 70-х (которым был Фортран, обладавший гораздо более существенными недостатками), Паскаль представлял собой значительный шаг вперёд. В начале 1980-х годов в СССР для обучения школьников основам информатики и вычислительной техники академик А. П. Ершов разработал алголо-паскалеподобный «алгоритмический язык».



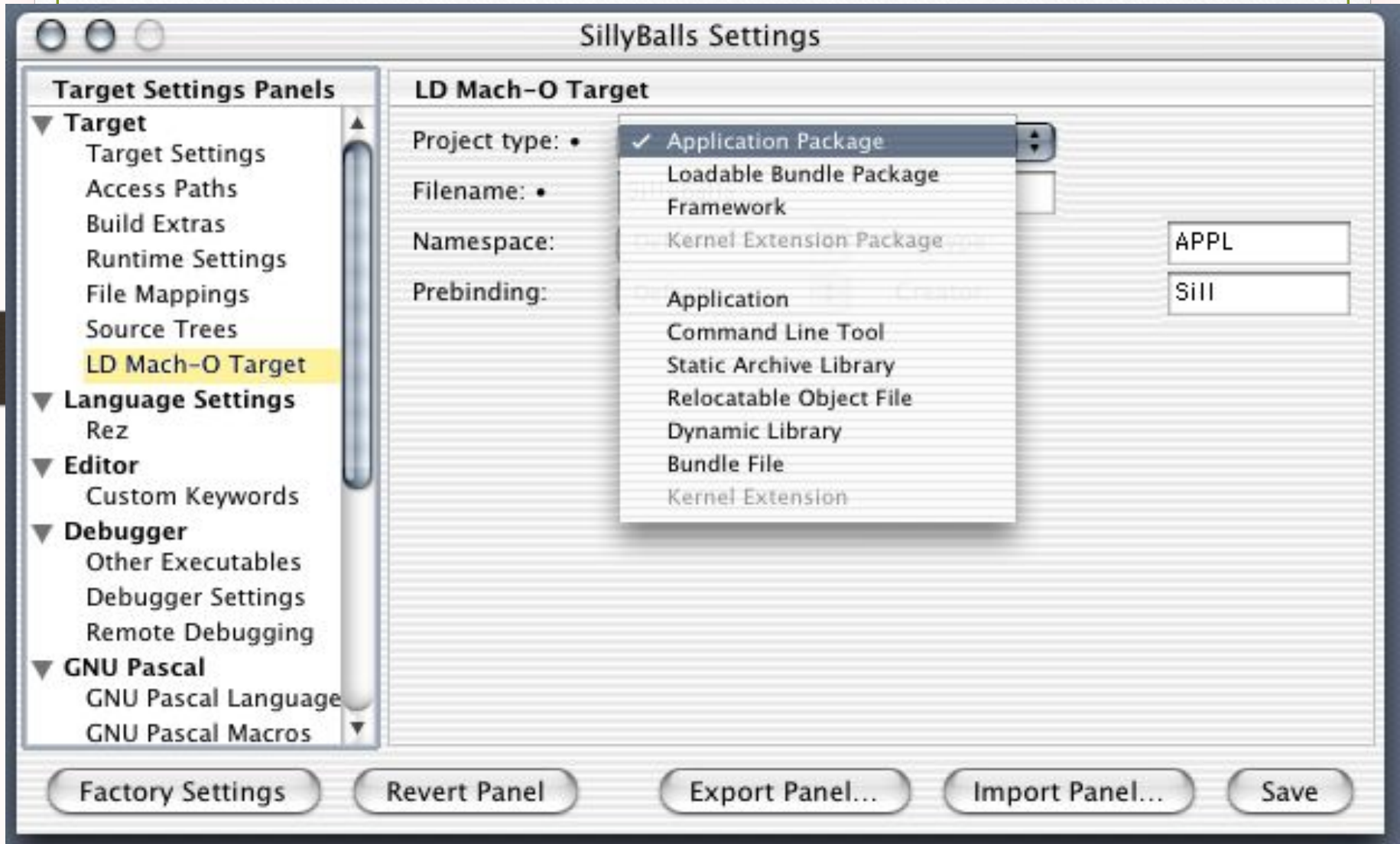
Pascal

Никлаус Вирт понимал недостатки созданного им языка, но, следуя традициям академической среды и собственным принципам, согласно которым «неподходящий инструмент надо не исправлять, а заменять», не стал его развивать дальше, а разработал новые языки семейства:

Модула-2 и Оберон. В противоположность этому промышленные традиции и достоинства языка побудили многие коммерческие и некоммерческие организации продолжать разрабатывать и развивать системы программирования именно на основе языка Паскаль, подвергая язык произвольному расширению, добавляя в него, часто совершенно механически, новые средства и синтаксические конструкции.

Наиболее известной реализацией Паскаля являлась система Turbo Pascal (выросшая затем в Borland Pascal для DOS/Windows и далее в Delphi) фирмы Borland, в которой использовались значительные расширения языка. Благодаря появлению развитых диалектов, язык стал богаче, но в отсутствие отраслевой стандартизации, потерял переносимость и общность (только в 1995 году появилась совместимая с Borland Pascal версия среды разработки Virtual Pascal для OS/2 и впоследствии Linux, в 1998 году Kylix — Delphi для Linux, в настоящее время оба этих проекта фактически заморожены).

Pascal



GNU Pascal

BASIC

BASIC — сокращение от англ. Beginner's All-purpose Symbolic Instruction Code — универсальный код символических инструкций для начинающих; англ. basic — основной, базовый) — семейство высокоуровневых языков программирования.

Был разработан в 1963 профессорами Дартмутского колледжа Томасом Куртцом (Thomas E. Kurtz, 1928-) и Джоном Кемени (John G. Kemeny, 1926—1993). Язык предназначался для обучения программированию и получил широкое распространение в виде различных диалектов, прежде всего, как язык для домашних микрокомпьютеров.

Бейсик был спроектирован так, чтобы студенты могли писать программы, используя терминалы с разделением времени. Он создавался как решение для проблем, связанных со сложностью более старых языков. Он предназначался для более «простых» пользователей, не столько заинтересованных в скорости программ, сколько просто в возможности использовать компьютер для решения своих задач.

BASIC

При проектировании языка использовались следующие восемь принципов, новый язык должен:

1. быть простым в использовании для начинающих
2. быть языком программирования общего назначения
предоставлять возможность расширения функциональности,
доступную опытным программистам
3. быть интерактивным
4. предоставлять ясные сообщения об ошибках
5. быстро работать на небольших программах
6. не требовать понимания работы аппаратного обеспечения
7. защищать пользователя от операционной системы

BASIC

Язык был основан частично на Фортран II и частично на Алгол-60, с добавлениями, делающими его удобным для работы в режиме разделения времени и, позднее, обработки текста и матричной арифметики. Первоначально Бейсик был реализован на мейнфрейме GE-265 с поддержкой множества терминалов. Вопреки распространённому убеждению, в момент своего появления это был компилируемый язык.

Взрывной рост

Несмотря на то, что язык уже использовался на нескольких миникомпьютерах, его настоящее распространение началось с его появления на микрокомпьютере Altair 8800. Многие языки программирования были слишком большими чтобы поместиться в небольшую память, которую пользователи таких машин могли себе позволить. Для машин с таким медленным носителем как бумажная лента (позднее — аудиокассета) и без подходящего текстового редактора такой небольшой язык как Бейсик был отличной находкой.

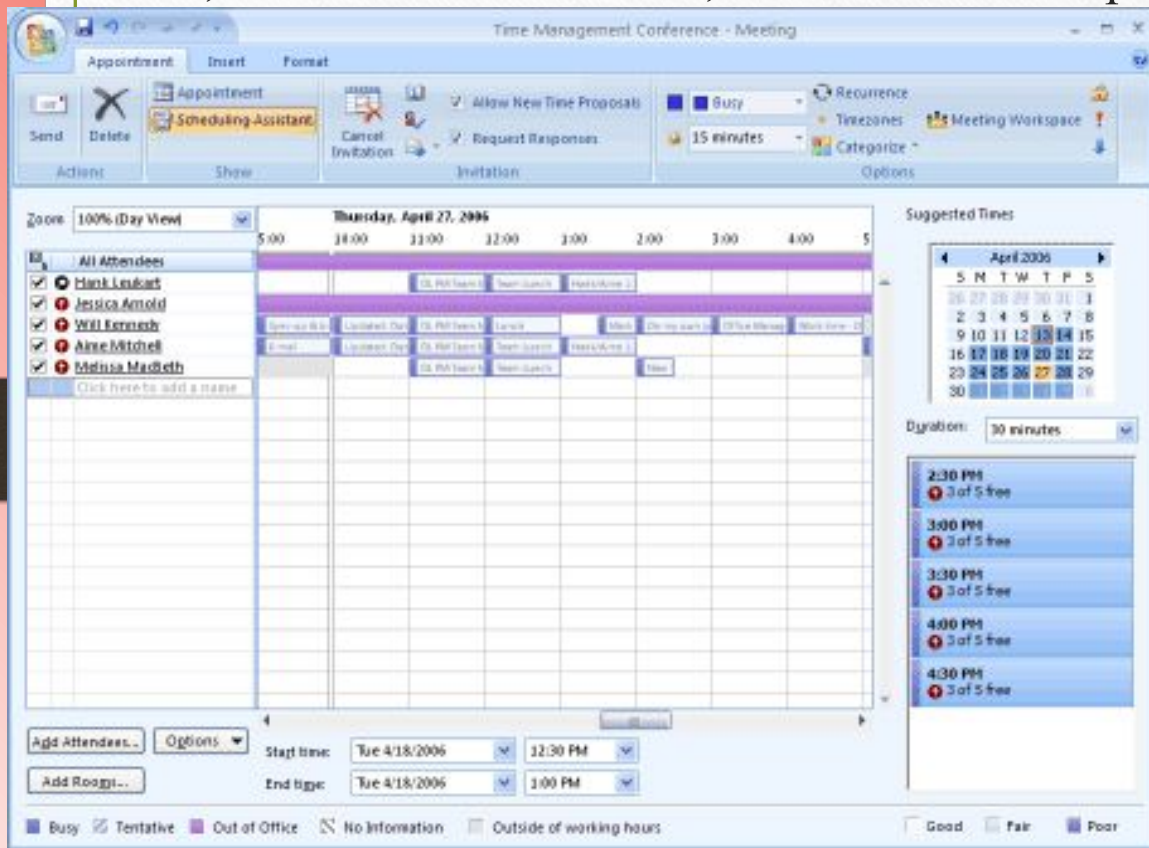
BASIC

Зрелость

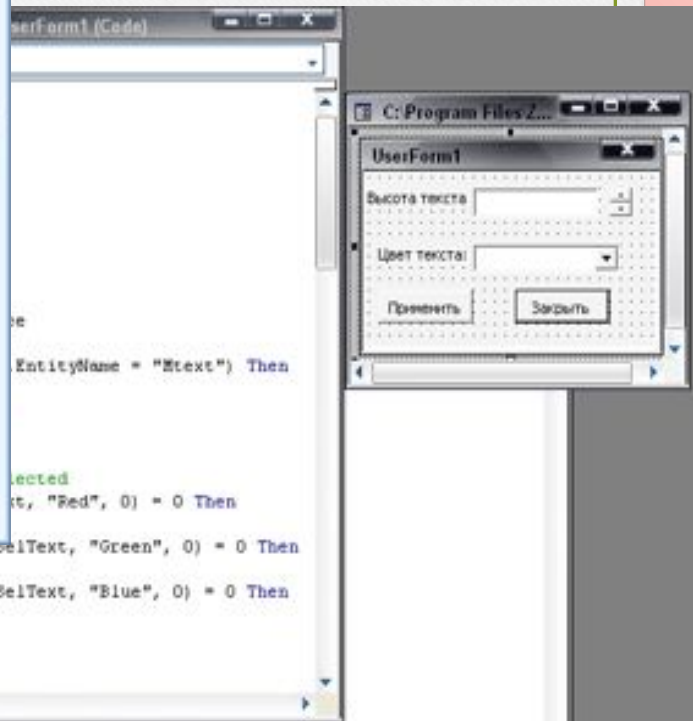
В этот период было создано несколько новых версий Бейсика. Майкрософт продавала несколько версий Бейсик для MS-DOS/PC-DOS, включая BASICA, GWBASIC (модификация BASICA, не требующая «прошивки» от IBM) и Quick BASIC (QBASIC). Borland, известная своим Turbo Pascal, в 1985 выпустила Turbo BASIC 1.0 (его наследники впоследствии продавались другой компанией под именем PowerBASIC). На домашних компьютерах появились различные расширения Бейсика, обычно включающие средства для работы с графикой, звуком, выполнением DOS-команд, а также средства структурного программирования. Некоторые другие языки использовали хорошо известный синтаксис Бейсика в качестве основы, на которой строилась совершенно иная система (см. например, GRASS).

BASIC

Однако, начиная с конца 80-х, новые компьютеры стали намного более удобными, как графический интерфейс, так и удобным для пользователя, несмотря на то, что BASIC все еще продавалось.



как графический интерфейс, так и удобным для пользователя, несмотря на то, что BASIC все еще продавалось.



интерпретатор Вейсика.

Microsoft Outlook

Бейсик используется в некоторых приложениях. Например, он встроен в отечественный калькулятор «**VB**Электроника МК-85».

Prolog

Пролог (Prolog) — язык логического программирования, основанный на логике предикатов первого порядка. Представляет собой подмножество логики предикатов первого порядка. Реализован в виде интерпретатора, который работает за счет ООП-архитектуры. Реализация Prolog в соответствии с ISO/IEC



The screenshot shows a window titled 'PROLOG.EXE' with a menu bar (Files, Edit, Run, Compile, Options, Setup) and a toolbar. The main editor area contains the following Prolog code:

```
clauses
  spisok([],0).
  spisok([X:Sp],Sum):-
    X>=0,
    spisok(Sp,Sum1),
    Sum=Sum1+X,!,
    spisok(Sp,Sum1),
    Sum=Sum1.!,.
```

Below the code, an error message is displayed: '403 Predicate name or section keyword expected.' The bottom of the window has a status bar with the following text: 'F1:Error explanation F2-Save F3-Load F5-Zoom F6-Next F10-Continue'.

Базовым типом данных являются списки. Записям в списках можно обратиться по индексу. Пролог не считает

Пролог отличается от обычных языков программирования (таких как C++, Java) в том, что установка баз данных оказываются более технологичными, чем аналогичные решения на Прологе. Негибкость заключается в трудности изучения языка, более высоких требований к квалификации программиста на Прологе, трудности отладки программы, неразвитости технологии программирования, плохой контролируемости промежуточных результатов.

е за счет
ые
ак
и
особы
темы
и

решения

C++

C++ представляет собой интересный эксперимент по адаптации возможностей объектной технологии к традиционному языку программирования. Бьерн Страуструп вполне достоин аплодисментов за то, что ему в голову пришла мысль слить обе технологии воедино. В то же время в C++ сохранились проблемы старого поколения средств программного производства. Язык C++ обладает тем преимуществом перед C, что поддерживает некоторые аспекты объектной технологии, которые могут быть использованы для ограниченного проведения анализа требований и проектирования. Однако процессы анализа, проектирования и реализации проекта все еще в значительной степени остаются внешними по отношению к C++. Таким образом, в C++ не реализованы важные преимущества объектной технологии, которые прямо бы привели к экономичному производству программной продукции.

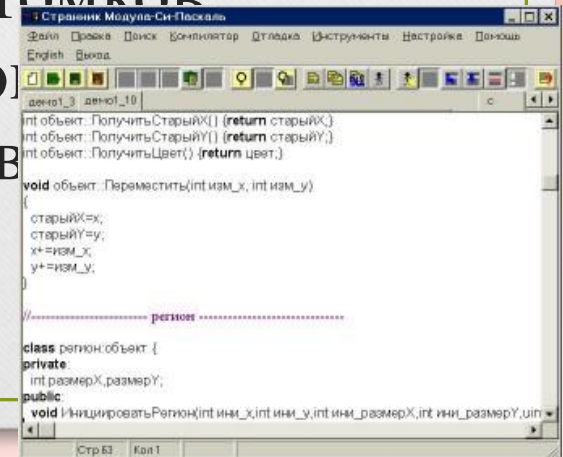
Полиморфизм - основополагающая концепция объектно-ориентированного программирования. В языке C++ ключевое слово `virtual` предоставляет функции возможность стать полиморфической, если она будет переопределена (переопределена) в одном классе-потомке или более. Однако слово `virtual` отнюдь не является необходимым, так как любая функция, переопределенная (`overridden`) в классе-потомке, может быть полиморфической. Компилятору только требуется генерировать коммутирующий код для истинно полиморфических процедур.

Если автор родительского класса в языке C++ не предвидит, что класс-потомок захочет переопределить функцию, то он не сможет сделать ее и полиморфической. В этом заключается наиболее серьезный порок C++, поскольку снижается гибкость программных компонентов, а следовательно, и способность создавать адаптируемые и расширяемые библиотеки.

C++ также позволяет функциям быть перегруженными (`overloaded`); в такой ситуации вызов нужной функции зависит от аргументов. Различие между перегруженными и полиморфическими (переопределенными) функциями состоит в том, что в перегруженных функциях нужная определяется при компиляции, а в случае полиморфических определяется при выполнении.

C++

Виртуальные функции предоставляют один из возможных путей реализации полиморфизма. Разработчик языка может сделать выбор в пользу определения полиморфизма либо в родительском, либо в наследующем классе. Так что же из них имеет смысл выбрать разработчику языка? Здесь можно выделить несколько вариантов для родительских классов и классов-потомков, которые не станут взаимоисключающими. Достаточно легко найти себе место в объектно-ориентированном языке.



```
Страница Модуль Си-Паскаль
Файл Правка Вставка Компилятор Отладка Инструменты Настройка Панель
English Выход
дене1_3 дене1_10
int объект: ПолучитьСтарыйX() {return старыйX;}
int объект: ПолучитьСтарыйY() {return старыйY;}
int объект: ПолучитьЦвет() {return цвет;}

void объект: Переместить(int изм_x, int изм_y)
{
    старыйX=x;
    старыйY=y;
    x+=изм_x;
    y+=изм_y;
}

//..... регион .....

class регион:объект {
private:
    int размерX, размерY;
public:
    void ИнициализироватьРегион(int ини_x, int ини_y, int ини_размерX, int ини_размерY, uint

```

SQL

SQL (англ. Structured Query Language - язык структурированных запросов) - универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Вопреки существующим заблуждениям, SQL является информационно-логическим языком, а не языком программирования.

SQL основывается на реляционной алгебре.

Целью разработки было создание простого непроцедурного языка, которым мог воспользоваться любой пользователь, даже не имеющий навыков программирования. Собственно разработкой языка запросов занимались Дональд Чэмбэрлин (Donald D. Chamberlin) и Рэй Бойс (Ray Boyce). Пэт Селинджер (Pat Selinger) занималась разработкой стоимостного оптимизатора (англ. cost-based optimizer), Рэймонд Лори (Raymond Lorie) занимался компилятором запросов.

Как и с многими стандартами, имеющими место в IT-индустрии, с языком SQL возникла проблема, что в прошлом многие производители использующего SQL ПО решили, что функционал в текущей (на тот момент времени) версии стандарта недостаточен (что, в принципе, для ранних версий SQL было в некотором роде справедливо) и его желательно расширить. Что и приводит в данный момент к тому, что у разных производителей СУБД в ходу разные диалекты SQL, в общем случае между собой несовместимые.

До 1996 года вопросами соответствия коммерческих реализаций SQL стандарту занимался в основном институт NIST, который и устанавливал уровень соответствия стандарту. Но позднее подразделение, занимавшееся СУБД, было расформировано, и на текущий момент все усилия по проверке СУБД на соответствие стандарту ложатся на её производителя.

Впервые понятие «уровня соответствия» было предложено в стандарте SQL-92. А именно, ANSI и NIST определяли четыре уровня соответствия реализации этому стандарту:

1. Entry (русск. базовый)
2. Transitional (русск. переходный) — проверку на соответствие этому уровню проводил только институт NIST
3. Intermediate (русск. промежуточный)
4. Full (русск. полный)

SQL

The image displays a collage of database management software interfaces. At the top, the word "SQL" is written in a large, stylized font. Below it, several windows are visible:

- SQL Server 2005 Enterprise Edition:** Shows the "AdventureWorks on SQLServer2005" server instance. The left pane shows a tree view of Schemas (6), Tables (6), Views (2), Triggers (6), and Indices (20). A "Data Wizard - Export Data" dialog box is open, showing a preview of a table with columns: Num, Name, Age.
- EMS MySQL Manager:** Shows the "dbdemos on mysql5server" instance. The left pane shows a tree view of Tables (12) and Views (4). The main pane shows a table named "biolife" with columns: Species_No, Category, Common_Name, Species_Name, Length_cm. The table contains 11 rows of data.
- SQL Server 2005 Query Builder:** Shows a table with columns: ContactID, NameStyle, Title, FirstName, MiddleName, LastName, Suffix, EmailAddress.
- SQL Server 2005 Query Editor:** Shows a SQL query:

```
SELECT * FROM `orders` WHERE `orders`.`CustNo` = :CustNo AND `orders`.`SaleDate` >= :SaleDateBegin AND `orders`.`SaleDate` <= :SaleDateEnd
```


Со времени создания первых программируемых машин человечество придумало уже более восьми с половиной тысяч языков программирования. Каждый год их число пополняется новыми. Некоторыми языками умеет пользоваться только небольшое число их собственных разработчиков, другие становятся известны миллионам людей. Профессиональные программисты иногда применяют в своей работе более десятка разнообразных языков программирования.



КОНЕЦ