



Ростовский Государственный Строительный Университет

Языки Программирования Высокого Уровня

Работу

выполнила

Папава Марика

Группа Э-108

Руководитель

Преподаватель
информатики

Клянина
Людмила

Николаевна

Введение

Создатели языков по-разному толкуют понятие язык программирования. К наиболее распространенным, предназначенным для записи программ, относятся следующие:

Программа обычно представляет собой некоторый алгоритм в форме, понятной для исполнителя (например, компьютера). Язык программирования определяет набор лексических, синтаксических и семантических правил, используемых при составлении компьютерной программы. Он позволяет программисту точно определить то, на какие события будет реагировать компьютер, как будут храниться и передаваться данные, а также какие именно действия

следует выполнять над этими данными при различных обстоятельствах. Язык выполняется над этими данными при различных обстоятельствах для определения и кодирования структурами данных и управления процессом вычислений.

Введение

Классы языков программирования:

1. Неклассифицированные языки
2. Императивные языки программирования
3. Структурные языки программирования
4. Процедурные языки программирования
5. Векторного программирования
5. Логические языки программирования
6. Аспектно-ориентированные
6. Программирование в ограничениях
7. Декларативные
8. Объектно-ориентированные языки программирования
8. Динамические
8. Функциональные языки программирования
10. Прототипные
10. Языки программирования для промышленной автоматизации
11. Объектно-ориентированные
12. Рефлексивные
10. Эзотерические языки программирования
13. Логического программирования
11. Стековые языки
14. Параллельного программирования
12. Параллельные языки 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 перевода программы в машинный код различна. В зависимости от того, как программа преобразовывается в машинный код, языки программирования делятся на компилируемые и интерпретируемые.

В компилируемых языках программа преобразовывается в машинный код заранее, до того, как программа будет исполнена. В интерпретируемых языках программа преобразовывается в машинный код по мере ее выполнения.

Программа на компилируемом языке при помощи специальной программы компилятора преобразуется (компилируется) в набор инструкций для компьютера (машинный код) и далее записывается в файл. Этот файл может быть запущен на выполнение как программа. Другими словами, компилятор переводит программу с языка высокого уровня на низкоуровневый язык, понятный процессору.

Если программа написана на интерпретируемом языке, то интерпретатор непосредственно выполняет (интерпретирует) ее текстовую форму. Интерпретатор переводит программу на исходном языке и не может быть запущена без интерпретатора. Можно сказать, что программа на интерпретируемом языке выполняется динамически, то есть во время ее выполнения.

Однако программы на интерпретируемом языке 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.

Cobool

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

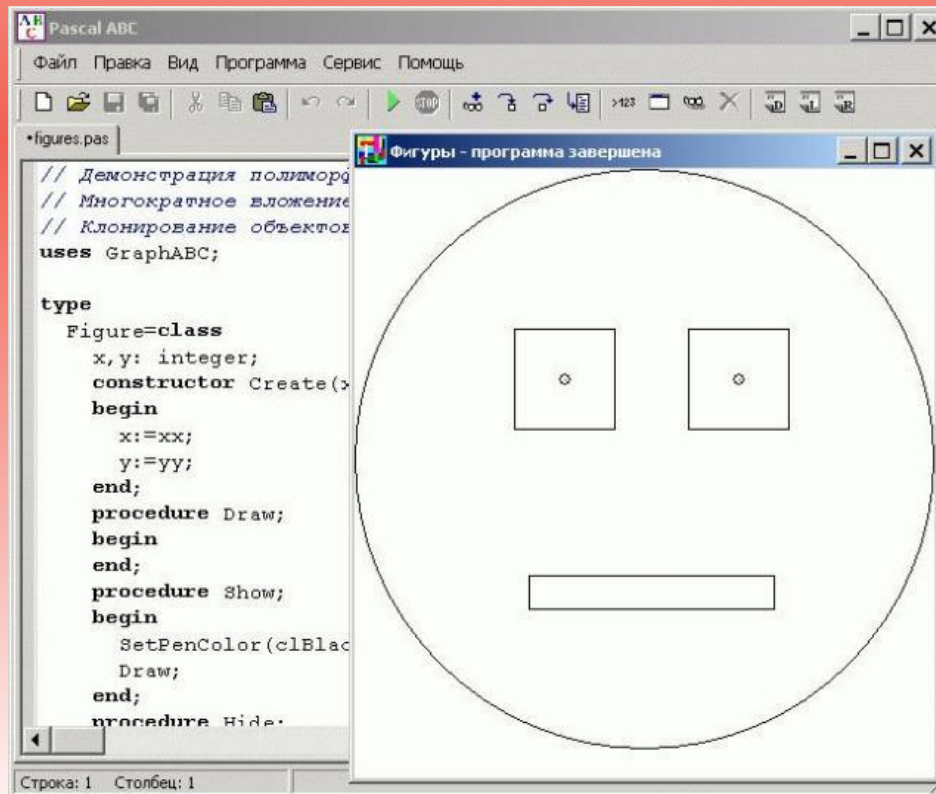
<< Back Next >> Finish

Algol

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.



The screenshot shows a Pascal ABC IDE window titled "Pascal ABC". The menu bar includes "Файл", "Правка", "Вид", "Программа", "Сервис", and "Помощь". The toolbar contains various icons for file operations and execution. The main window is split into two panes. The left pane shows the source code for a file named "figures.pas". The code includes comments in Russian and Pascal code defining a class-like structure for drawing a smiley face. The right pane shows the output window titled "Фигуры - программа завершена", which displays a simple smiley face drawn with black lines on a white background. The status bar at the bottom indicates "Строка: 1 Столбец: 1".

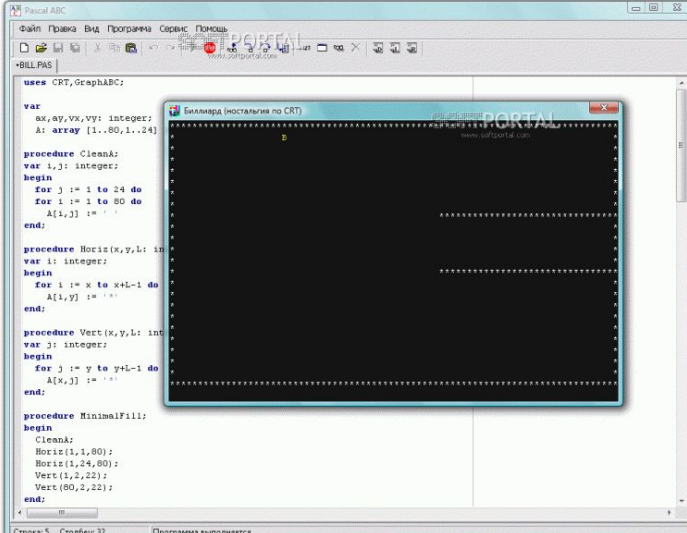
```
// Демонстрация полиморфизма
// Многократное вложение
// Клонирование объектов
uses GraphABC;

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;
```


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».



```
uses CRT, GraphABC;

var
  ax, ay, vx, vy: integer;
  A: array [1..80, 1..24];

procedure CleanA;
var i, j: integer;
begin
  for j := 1 to 24 do
  for i := 1 to 80 do
  A[i, j] := ' ';
end;

procedure Horiz(x, y, l: integer);
var i: integer;
begin
  for i := x to x+l-1 do
  A[i, y] := ' ';
end;

procedure Vert(x, y, l: integer);
var j: integer;
begin
  for j := y to y+l-1 do
  A[x, j] := ' ';
end;

procedure MinimalFill;
begin
  CleanA;
  Horiz(1, 1, 90);
  Horiz(1, 24, 80);
  Vert(1, 2, 22);
  Vert(80, 2, 22);
end;

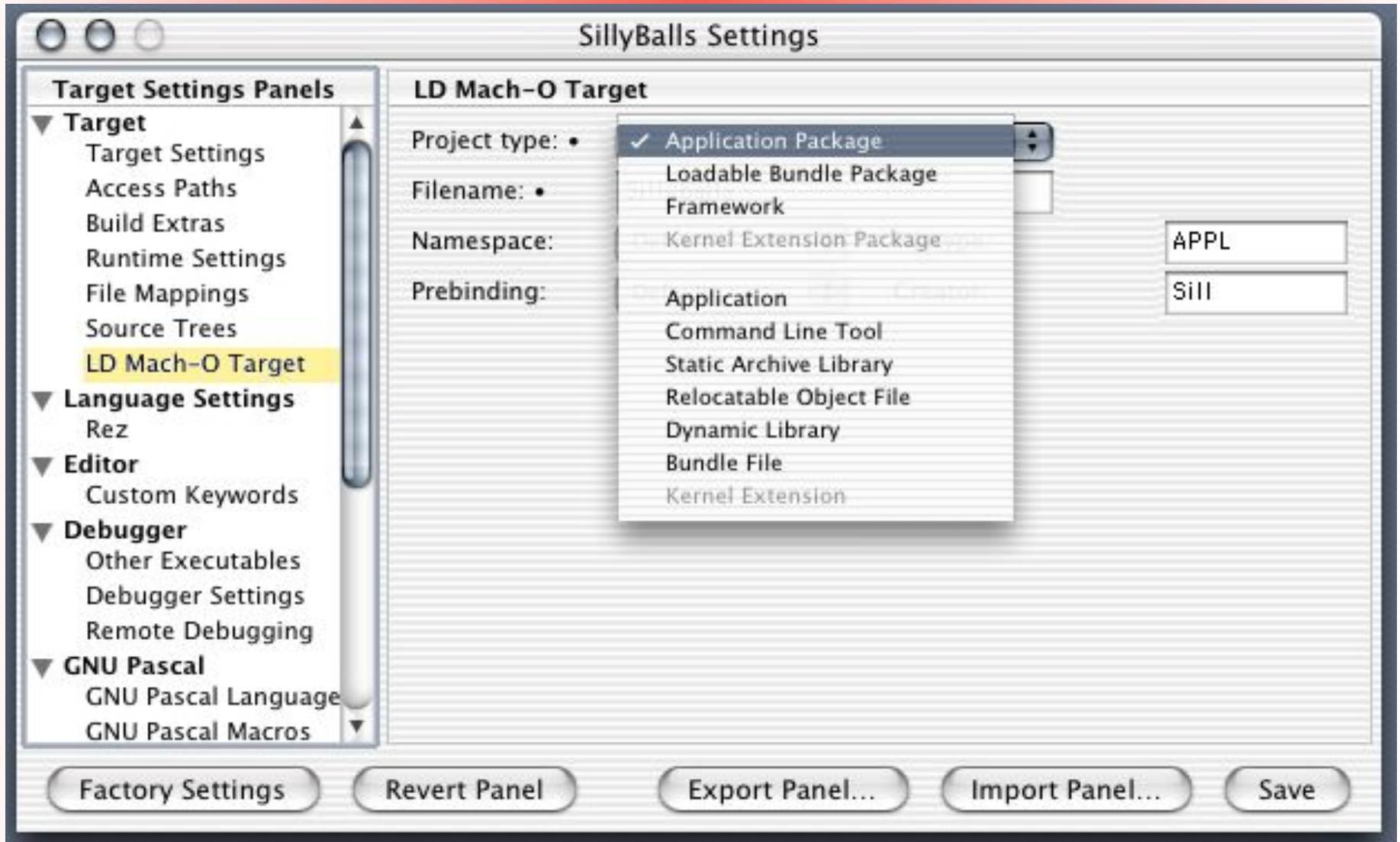
MinimalFill;
```

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.

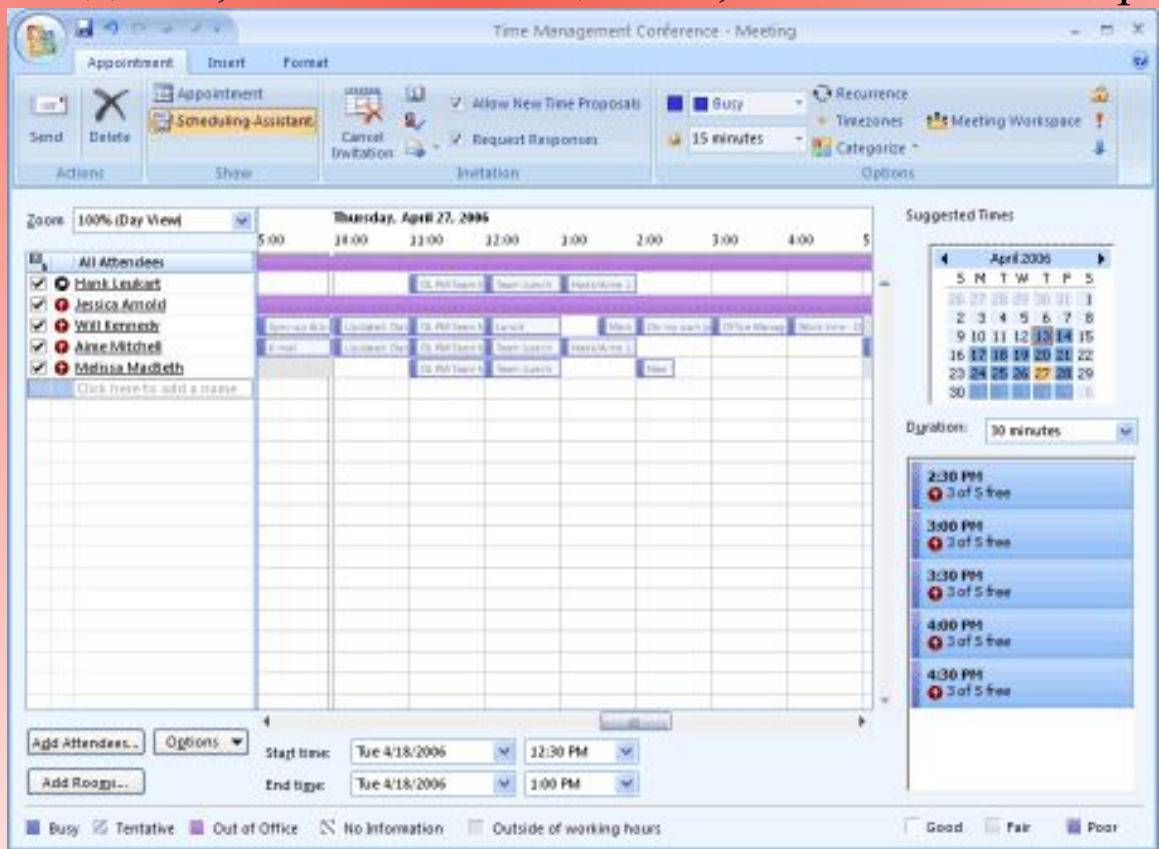
BASIC

Зрелость

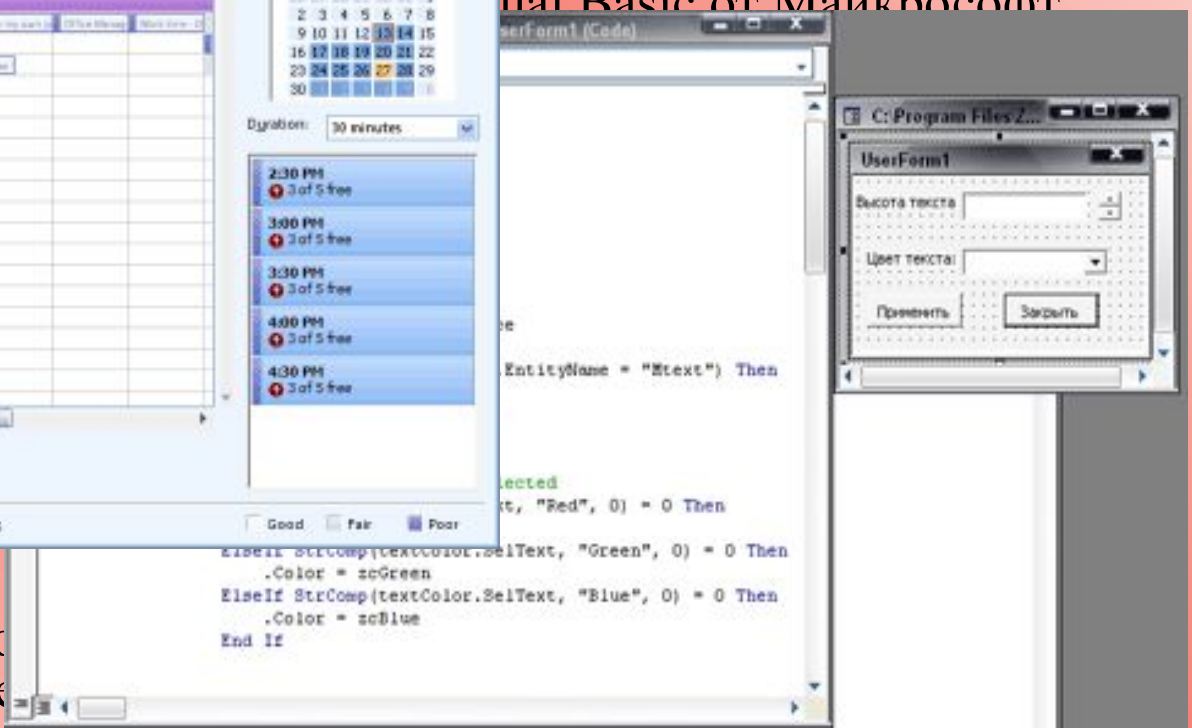
В этот период было создано несколько новых версий Бейсика. Майкрософт продавала несколько версий Бейсик для MS-DOS/PC-DOS, включая BASICA, GWBASIC (модификация BASICA, не требующая «прошивки» от IBM) и Quick BASIC (QBASIC). Borland, известная своим Turbo Pascal, в 1985 выпустила Turbo BASIC 1.0 (его 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 как графический интерфейс роль удобным для позиции, несмотря на то, что лось и продавалось.



Visual Basic от Майкрософт



Microsoft Outlook
Бейсик используется в некоторых приложениях.
Например, он встроен в от

VBA

Prolog

Пролог (Prolog) — язык логического программирования, основанный на логике

дизъюнкторов Хорна, представляющей собой подмножество логики предикатов первого

порядка. Пролог реализован в виде интерпретатора, работающего за счет

ООП-архитектуры. Пролог реализован в виде интерпретатора, работающего за счет

реализации стандарта ISO/IEC 13211-1. Пролог реализован в виде интерпретатора, работающего за счет

ISO/IEC 13211-1. Пролог реализован в виде интерпретатора, работающего за счет

Базовые данные. Пролог реализован в виде интерпретатора, работающего за счет

данных. Пролог реализован в виде интерпретатора, работающего за счет

записей. Пролог реализован в виде интерпретатора, работающего за счет

их обра. Пролог реализован в виде интерпретатора, работающего за счет

Пролог реализован в виде интерпретатора, работающего за счет

считаю. Пролог реализован в виде интерпретатора, работающего за счет

Пролог реализован в виде интерпретатора, работающего за счет

решени. Пролог реализован в виде интерпретатора, работающего за счет

данных. Пролог реализован в виде интерпретатора, работающего за счет

Негибкость заключается в трудности изучения языка, более высоких требований к

квалификации программиста на Прологе, трудности отладки программы, неразвитости

технологии программирования, плохой контролируемости промежуточных результатов.



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.

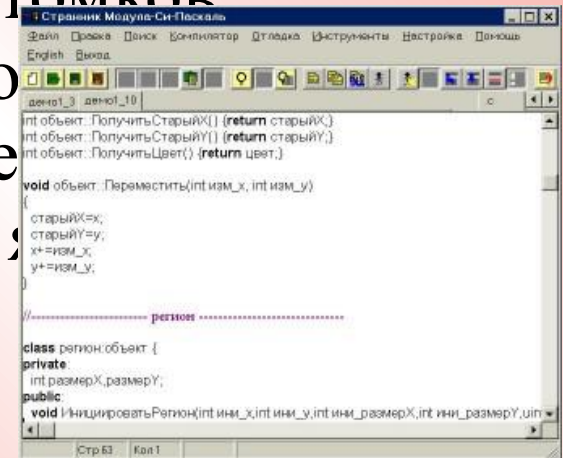
Полиморфизм - основополагающая концепция объектно-ориентированного программирования. В языке C++ ключевое слово `virtual` предоставляет функции возможность стать полиморфической, если она будет переписана (переопределена) в одном классе-потомке или более. Однако слово `virtual` отнюдь не является необходимым, так как любая функция, переопределенная (`overriden`) 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 смогут достаточно легко найти себе любом объектно-ориентированном



```
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, uin
Стр 63 Кол 1
```

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

SQL

The image displays a composite of several software windows related to database management. At the top, the title 'SQL' is prominently shown in a large, stylized font. Below it, the main interface is divided into several overlapping windows:

- SQL Server 2005 Enterprise Edition:** Shows the 'AdventureWorks on SQLServer2005' server instance. The left pane displays a tree view of the database structure, including Schemas (dbo, HumanResources, Person), Tables (Address, AddressType, Contact, ContactType, CountyRegion, StateProvince), Views (Procedures, Triggers, Indices), and other objects. The right pane shows the 'Object Explorer' for the 'AdventureWorks' database, displaying the 'dbo' schema and various tables like 'Address' and 'Contact'.
- EMS MySQL Manager:** Shows the 'dbdemos on mysql5server' instance. The left pane displays a tree view of the database structure, including Tables (biolife, county, custody, customer, employee, items, mymapoints, orders, parts, reservat, vendors, veruies) and Views (4). The right pane shows the 'Table - [biolife]' window, displaying a list of fish species with columns: Species_No, Category, Common_Name, Species_Name, and Length_cm. The data is as follows:

Species_No	Category	Common_Name	Species_Name	Length_cm
90020	Triggerfish	Clown Triggerfish	Ballistodes conspicillum	50,000
90030	Snapper	Red Emperor	Lutjanus sebae	60,000
90050	Wrasse	Giant Maori Wrasse	Cheilinus undulatus	229,000
90070	Angelfish	Blue Angelfish	Pomacanthus nauarchus	30,000
90080	Cod	Lunartail Rockcod	Variola louti	90,000
90090	Scorpionfish	Firefish	Pterois voltans	38,000
90100	Butterflyfish	Ornate Butterflyfish	Chaetodon Ornatissimus	19,000
90110	Shark	Swell Shark	Cephaloscyllium ventriosum	102,000
- Export Data Wizard:** A dialog box titled 'Data Wizard - Export Data' is open, showing 'Customize HTML export options'. The 'Preview' tab is selected, displaying a table with columns 'Num', 'Name', and 'Age'. The data is as follows:

Num	Name	Age
1	John	34
2	Marcella	27
3	Alex	25
4	Julia	48
- Query Builder:** A window titled 'Query Builder - [AdventureWorks]' is visible, showing 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

Вопросы

1. Что такое язык программирования? (сл.2)
2. Функции языков программирования? (сл.2)
3. Их исполнение? (сл.2)
4. Может ли язык программирования использовать специальные конструкции для определения и манипулирования структурами данных? (сл.2)
5. Классы языков программирования? (сл.3)
6. Перечислите список языков программирования. (сл.3)
7. К какому языку программирования относится Basic? (сл.5)
8. Назовите язык процедурного программирования. (сл.6)
9. В каком виде сохраняются данные в современных цифровых компьютерах? (сл.9)
10. Перечислите основные структуры данных. (сл.10)

Вопросы

11. Что делает Деривационная семантика? (сл.11)
12. Чем оперирует Денотационная семантика? (сл.11)
13. Можно ли создать для интерпретируемого языка компилятор? (сл.13)
14. Приведите примеры. (сл.13)
15. Какие программы работают быстрее, компилируемые или интерпретируемые? (сл.13)
16. Что такое язык высокого уровня? (сл.14)
17. Для чего он предназначен? (сл.14)
18. Перечислите хотя бы 5 языков высокого уровня.
19. Первый реализованный язык программирования высокого уровня? (сл.15)
20. Назовите язык программирования предназначенный для разработки бизнес-приложений? (сл.16)

Вопросы

21. Какой язык получил распространение, как язык для домашних компьютеров? (сл.23)
22. Принципы использовавшиеся при проектировании языка Basic? (сл.24)
23. Язык логического программирования? (сл.28)
24. Базовый принцип Prolog(a)? (сл.28)
25. Его недостатки? (сл.28)
26. Что такое полиморфизм? (сл.29)
27. На чём основывается язык SQL? (сл.31)