Paradigmas de Programação

Histórico das linguagens de programação

Sumário

- Conceito
- Paradigmas
- Abstrações
- Linguagens de Scripting
- Histórico
- Detalhes das Linguagens de Programação

Conceito

- Linguagem de Programação é um sistema de notações para descrição de computação sob forma legível tanto para o ser humano quanto para as máquinas
- A descrição de Computação pode ser feita de diversas maneiras:
  - Pela Programação imperativa – O estado do sistema é repetidamente atualizado por atribuições.
  - Pela Programação funcional – Novos estados são criados repetidamente em vez de atualizar estados anteriores.
  - Pela Programação lógica – Estabelece-se o que seja considerado verdade sobre uma solução e a linguagem busca uma solução.

Conceito

- Para estabelecer o que significa legível para o ser humano é preciso afirmar que o ser humano não precisa conhecer perfeitamente o funcionamento do computador
- Para fazer uma analogia com os automóveis, basta lembrar que não há necessidade de conhecer seu funcionamento interno para dirigi-los
- A maioria dos motoristas não tem noção do funcionamento dos pistões engrenagens, velas, injetores, radiadores, etc
- A direção é possível com o conhecimento de pequeno número de abstrações:
  - Ignição – inserir a chave e gira-la faz o caro dar partida.
  - Acelerador – pressiona-lo faz o carro acelerar.
  - Volante – gira-lo faz o movimento do carro mudar de direção.
  - Freio – pressiona-lo faz o carro parar.

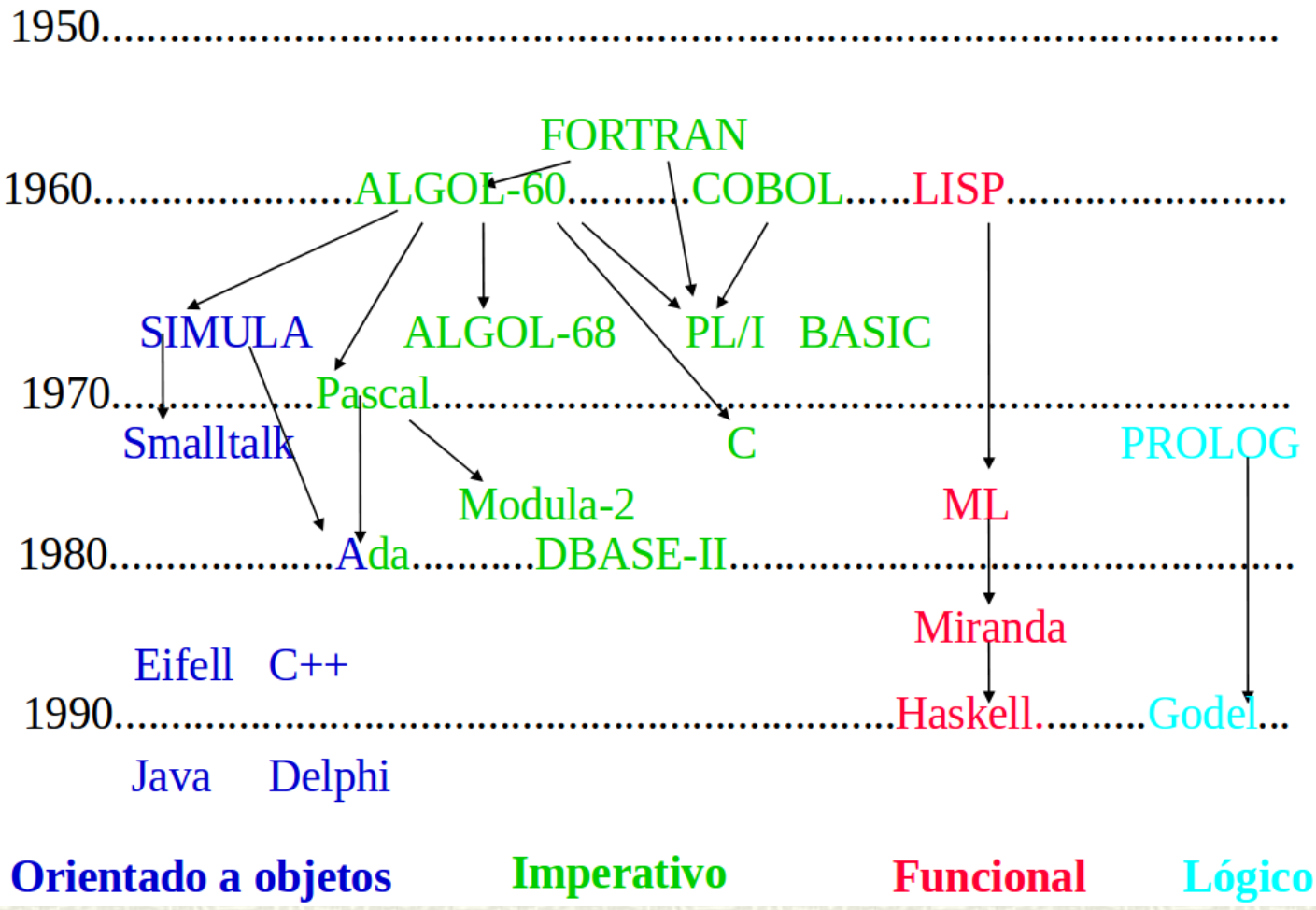
Conceito

- Para definir uma abstração é definido um pequeno número de operações na abstração que de forma alguma fornece o código interno de implementação da abstração
- Estas abstrações podem ser de controle e de dados.
- Linguagens que preservam as abstrações, não importando as operações que o programa esteja executando são chamadas “seguras” ou “fortemente tipadas”
- Aquelas nas quais o programa pode violar as abstrações são consideradas “não seguras” ou fracamente tipadas.
- A história das linguagens de programação tem se baseado no desenvolvimento e refinamento das abstrações
- Entre 1950 e 1970 o foco de desenvolvimento era nas abstrações de controle enquanto a partir de 1970 as abstrações de dados passaram a receber maior atenção.

Paradigmas

- Um paradigma é essencialmente um modo, de alto nível, de especificar sobre o que é a computação
- Isto é o fator de unificação que faz as linguagens Ada e C serem bastante similares, ignorando detalhes tais como as estruturas de controle, passagem de parâmetros, etc
- Alguns paradigmas existentes são:
  - Imperativo – Fortan, Cobol, Pascal, C, Basic
  - Funcional – LISP
  - Lógico – Prolog
  - Paralelo - OpenMP

Histórico


1950
FORTRAN
1960
ALGOL-60
COBOL
LISP
SIMULA
ALGOL-68
PL/I
BASIC
1970
Pascal
Smalltalk
C
PROLOG
Modula-2
ML
1980
Ada
DBASE-II
Miranda
Eifell
C++
1990
Haskell
Godel
Java
Delphi
Orientado a objetos
Imperativo
Funcional
Lógico

Características das Linguagens de Programação

Fortran
LISP
ALGOL
COBOL
BASIC
PL/I
Snobol
Simula 67
ALGOL 68
Pascal
C
JCL
Rexx
WFL

"Shells" Unix (sh, csh, ksh, ...)
Prolog
Ada
Smalltalk
C++
Perl
Tcl
Python
C#
Visual Basic (VB)
Delphi
Java
C++ Builder

Clique para adicionar o título

Características das Linguagens de Programação

Click to add Text

FORTRAN

- Formula Translation foi criada para atender cientistas e engenheiros

Versão	Data	Característica
0	1954	Identificadores com dois caracteres
I	1957	Identificadores com seis caracteres, if aritmético, teste no final, tipos implícitos pela inicial (John Backus e mais 17 pessoas)
II	1958	Compilação separada de rotinas
IV	1962	Declaração de tipos, if lógico
77	1978	Strings de caracteres, else
90	90	“arrays” dinâmicos, registros, ponteiros, case, recursão, módulos

FORTRAN (exemplo)

```
C FORTRAN77 PROGRAM TO FIND MEAN OF N NUMBERS AND NUMBER OF
C VALUES GREATER THAN MEAN (PS THIS IS TYPICAL, RATHER THAN
C QUALITY, FORTRAN CODE)
DIMENSION A(99)
REAL MEAN
READ (1,5) N
5 FORMAT(I2)
READ(1,10) (A(I),I=1,N)
10 FORMAT(6F10.5)
SUM = 0.0
DO 15 I = 1,N
15 SUM = SUM + A(I)
MEAN = SUM / FLOAT(N)
NUMBER = 0
DO 20 I = 1,N
IF (A(I) .LE. MEAN) GOTO 20
NUMBER = NUMBER + 1
20 CONTINUE
WRITE(2,25) MEAN, NUMBER
25 FORMAT(8H MEAN = ,F10.5,5X,20H NUMBER OVER MEAN = ,I5)
STOP
END
```

LISP

- 1959
- Primeira linguagem a suportar processamento de listas
- Primeira linguagem a suportar recursão
- Primeira linguagem funcional
- Variáveis representam parâmetros e não posições de memória

LISP (exemplo)

```
(DEFUN POT (X Y)
   (DO ((RESULTADO 1)
   ; ligação e atribuição de parâmetros
            (EXPOENTE Y))
   ; ligação e atribuição de parâmetros
            ((ZEROP EXPOENTE) RESULTADO)
   ; Teste e retorno
            (SETQ RESULTADO (* X RESULTADO))
    ; Corpo
            (SETQ EXPOENTE (- EXPOENTE 1))))
    ; Corpo
```

ALGOL

- 1960
- Formalização dos tipos de dados
- Identificadores de quaisquer comprimento
- “Arrays” além de 3 dimensões
- Estrutura de blocos
- Procedimentos recursivos
- Tremendo sucesso como linguagem de especificação
- Primeira linguagem formalmente especificada – BNF
- Forte influencia sobre outras linguagens – Ada, C, Pascal

ALGOL (exemplo)

```
begin comment this program is the ALGOL 60 version of finding the mean and
      the number of those greater than the mean
integer n;
begin real array a[1:n];
integer i, number; real sum, mean;
for i:=1 step 1 until n do read(a[i])
sum:=0.0;
for i:=1 step 1 until n do
sum := sum+a[i];
mean := sum/n;
number:=0;
for i:=1 step 1 until n do
if a[i] > mean then number := number+1;
write("MEAN =",mean,"NUMBER OVER MEAN =",number)
end
end
```

COBOL

- 1961
- Especificação do Departamento de Defesa norte americano
- Gigantesca penetração no mercado, só comparável a do Fortran

- Programa estruturado em quatro divisões
- IDENTIFICATION DIVISION
- ENVIRONMENT DIVISION
- DATA DIVISION
- PROCEDURE DIVISION

- As divisões sub dividem-se em parágrafos.
- O separador é o “.”
- Tipos extremamente detalhados
- Fragilidade de sub programação

COBOL (exemplo)

```
IDENTIFICATION DIVISION.
PROGRAM-ID INOUT.
*This program reads in one file, extends each record,
*and writes out a new file
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
SELECT INP-FIL ASSIGN TO INFILE.
SELECT OUT-FIL ASSIGN TO OUTFILE.
DATA DIVISION.
FILE SECTION.
FD INP-FIL
LABEL RECORDS STANDARD
DATA RECORD IS REC-IN.
01 REC-IN.
05 ALPHA-IN PIC A(4).
05 SP-CH-IN PIC X(4).
05 NUM-IN PIC 9(4).
FD OUT-FIL
LABEL RECORDS STANDARD
DATA RECORD IS REC-OUT.
01 REC-OUT.
05 ALPHA-OUT PIC A(4).
05 SP-CH-OUT PIC X(4).
05 NUM-OUT PIC 9(4).
05 EXTRAS PIC X(16).
WORKING STORAGE SECTION.
01 EOF PIC X VALUE IS 'N'.
PROCEDURE DIVISION.
AA.
OPEN INPUT INP-FIL
OPEN OUTPUT OUT-FIL
PERFORM CC
PERFORM BB THRU CC UNTIL EOF = 'Y'
CLOSE INP-FIL, OUT-FIL
DISPLAY "End of Run"
STOP RUN.
BB.
MOVE REC-IN TO REC-OUT
MOVE 'EXTRA CHARACTERS' TO EXTRAS
WRITE REC-OUT.
CC.
READ INP-FIL
AT END MOVE 'Y' TO EOF.
```

BASIC

- Lançada em 1964 por alunos de pós graduação (John G. Kemeny e Thomas E. Kurtz) do Dartmouth College
- Corruptela da linguagem Fortran
- Nas versões iniciais de Basic cada linha era numerada e esse números eram utilizados como rótulos para desvios.
- Imaginava-se que, tal como para uma calculadora, cada programa teria entre 10 a 15 linhas de código.
- Com o advento dos microcomputadores, em 1975, as linguagens de programação tradicionais não podiam ser utilizada por necessitarem de compiladores pesados e exigirem memória então indisponível ou até impensável para a época
- Bill Gates desenvolveu a primeira linguagem de terceira geração para micros, o BASIC
- BASIC tornou-se a opção preferencial nos microcomputadores
- Ampla disseminação dos micros deu origem a aplicativos de uso geral cujo código era completamente ilegível por parte de quem não houvesse participasse de seu desenvolvimento.
- A Microsoft desenvolveu uma versão denominada Quick Basic.
- Essa linguagem recebeu pouco alento até Alan Cooper criar o Visual Basic

BASIC (exemplo)

```
10 REM THIS IS A BASIC PROGRAM FOR FINDING THE MEAN
20 DIM A(99)
30 INPUT N
40 FOR I = 1 TO N
50 INPUT A(I)
60 LET S = S + A(I)
70 NEXT I
80 LET M = S/N
90 LET K = 0
100 FOR I = 1 TO N
110 IF A(I) < M THEN 130
120 LET K = K + 1
130 NEXT I
140 PRINT "MEAN IS", MEAN
150 PRINT "NUMBER GREATER THAN MEAN IS", K
160 STOP
170 END
```

PL/I

- 1960
- Linguagem criada pela empresa IBM para ser uma linguagem geral “para todas as estações” pretendendo substituir simultaneamente o Fortran e o Cobol.

PL/I (exemplo)

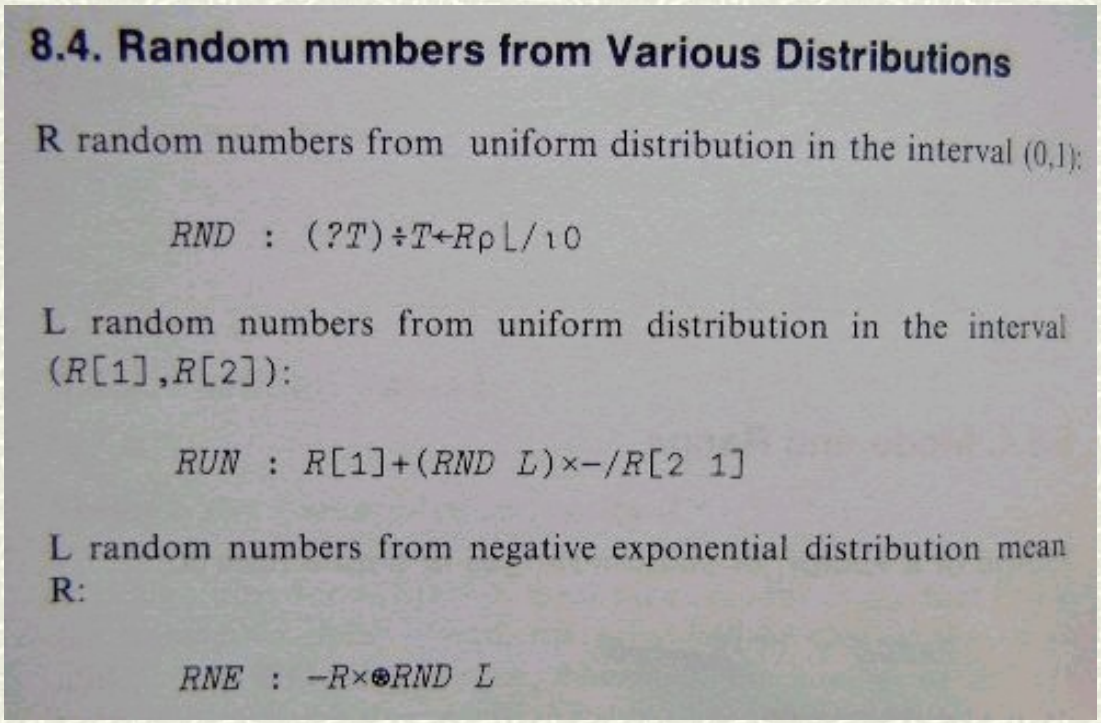
```
EXAMPLE : PROCEDURE OPTIONS (MAIN);
 /* PL/I version of the means program */
GET LIST (N);
 IF N > O THEN BEGIN;
 DECLARE MEAN, A(N) DECIMAL FLOAT,   SUM DEC FLOAT INITIAL(0), NUMBER
      FIXED INITIAL(0);
 GET LIST (A);
 DO I = 1 TO N;
  SUM = SUM + A(I)
 END;
 MEAN = SUM/N;
 DO I = 1 TO N;
   IF A(I) > MEAN THEN
     NUMBER = NUMBER + 1;
  END;
 PUT LIST ('MEAN=', MEAN, 'NUMBER GREATER THAN MEAN=', NUMBER);
END EXAMPLE;
```

APL

- A linguagem APL (“A Program Language”) foi criada por Keneth Iverson quando publicou, em Stanford em 1962, um livro de mesmo nome
- Em 1966 foi implementada a linguagem APL360 para “mainframes” IBM
- Linguagem matricial e foi a primeira linguagem de interpretação a tratar “arrays” como uma entidade
- Baseava-se fortemente em símbolos únicos para a representação de funções, enquanto outras linguagens usam cadeias de caracteres para representar as funções
- Usa símbolos algébricos comuns, letras gregas e outros símbolos para representar funções primitivas e operadores.
- Programas escritos em APL são extremamente concisos utilizando cerca de 20 a 30 vezes menos linhas de código que outras linguagens.

APL (exemplo)

```
VEC←15 3.141 888 99.98 34567
(+/,VEC)÷⍴,VEC
7114.6242
```



8.4. Random numbers from Various Distributions

R random numbers from uniform distribution in the interval (0,1):

```
RND : (?T)÷T←R⍴⌊/⍳0
```

L random numbers from uniform distribution in the interval (*R*[1],*R*[2]):

```
RUN : R[1]+(RND L)×-/R[2 1]
```

L random numbers from negative exponential distribution mean R:

```
RNE : -R×⍟RND L
```

APL (exemplo)

```
      ∇ L NEWRAP R;I;T;T1;X
[1]   T←(¯1+R⍳';')↑R
[2]   T1←(R⍳';')↓R
[3]   X←L[3]
[4]   I←0
[5]  L1:Z←X-(⌹T)÷⌹T1
[6]   →L3 IF L[1]>|X-Z
[7]   →L2 IF 0≠L[2]|I←I+1
[8]   'AT ITERATION ',(⍕I),' X = ',(3↓L)⍕X
[9]  L2:X←Z
[10]  →L1
[11]  L3:'SOLUTION = ',((3↓L)⍕X),
                      ' AT ITERATION ',⍕I
      ∇
```

Snobol

- StriNg Oriented symbOlic Language ou Snobol é uma linguagem de uso específico desenvolvida entre 1962 e 1967 nos Bell Laboratories por Farber, Griswold e Polensky para a manipulação de cadeias de caracteres
- Sua aplicação mais comum foi a geração de editores de texto
- Atualmente é pouco usada, tendo sido substituída por Perl.

Snobol (exemplo)

```
* BLANK IS A CONCATENATION OPERATOR!
        FIRST = 'HORACE'
        SECOND = 'GREELEY'
        NAME = FIRST ' ' SECOND
        OUTPUT = NAME
* MULTIPLICATION HAS HIGHER PRECEDENCE THAN DIVISION
* ARITHMETIC HAS HIGHER PRECENCE THAN CONCATENATION
        OUTPUT = '12 / 2 * 3 = ' 12 / 2 * 3
* CONCATENATE A STRING AND A NUMBER
        ROW = 'K'
        NO. = 23
        SEAT = ROW NO.
        OUTPUT = SEAT
* GET AN INPUT STRING
        STUFF = INPUT
* CHANGE ALL INSTANCES OF 'CAT' TO 'DOG"
LOOP                STUFF 'CAT' = 'DOG' :S(LOOP)
        OUTPUT = STUFF
* IS IT A NUMBER OR A STRING?
        X = 1234
        X '2' = 5
        OUTPUT = X
END
```

Simula67

- 1967
- Descendente do Algol 60, tendo antes passado pela Simula 1
- Foi uma linguagem importante a introduzir:
  - Co-rotinas – sub programas que podem reiniciar no ponto aonde foram interrompidos
  - Objetos – empacotamento de estruturas de dados e rotinas
  - Construtor de classes – deixando a criação de instâncias para o momento oportuno
  - Herança de classes – permitindo o desenvolvimento de hierarquias de classes
- Simula 67 teve limitada importância direta em virtude da limitação de seu domínio de aplicação
- Todavia sua influencia em linguagens futuras foi enorme.

Simula67 (exemplo)

```
Begin
    while 1 = 1 do begin
        outtext ("Hello World!");
        outimage;
    end;
End;
```

Mumps

- A linguagem Mumps ou M é originada de “Massachussetts General Hospital Utility Multi-Programming System” que era um sistema operacional completo com uma linguagem de programação para emprego nos mini computadores de 1967. A linguagem tinha por características:
  - Ser procedimental
  - Interpretada
  - Para uso geral
  - Com variáveis em tipo, convertendo-se automaticamente entre caracteres e tipos numéricos
  - Com “arrays” multi-dimensionais e associativos
  - Com variáveis persistentes ou globais
  - Com boa capacidade de tratamento de cadeias de caracteres
  - Com capacidade de múltiplos usuários e multitarefa
  - Com capacidade de indireção computando valores de “strings” em tempo de execução

Mumps (exemplo incompleto)

```
%DTC
 %DTC ; SF/XAK - DATE/TIME OPERATIONS ;1/16/92 11:36 AM
          ;;19.0;VA FileMan;;Jul 14, 1992
          D       I   'X1!'X2 S X="" Q
          S X=X1 D H S X1=%H,X=X2,X2=%Y+1 D H S X=X1-%H,%Y=%Y+1&X2
         K %H,X1,X2 Q
          ;
 C       S X=X1 Q:'X D H S %H=%H+X2 D YMD S:$P(X1,".",2) X=X_"."_$P(X1,".",2)
 K X1,X2 Q
 S       S %=%#60/100+(%#3600\60)/100+(%\3600)/100 Q
          ;
 H        I  X<1410000 S %H=0,%Y=-1 Q
          S %Y=$E(X,1,3),%M=$E(X,4,5),%D=$E(X,6,7)
          S %T=$E(X_0,9,10)*60+$E(X_"000",11,12)*60+$E(X_"00000",13,14)
TOH    S
%H=%M>2&'(%Y#4)+$P("^31^59^90^120^151^181^212^243^273^304^334","^",%M)+%D
        S %='%M!'%D,%Y=%Y-141,%H=%H+(%Y*365)+(%Y\4)-(%Y>59)+%,%Y=$S(%:- 1,1:%H+4#7)
```

Algol68

- 1968
- A continuação do esforço do Algol 60 levou à criação de uma linguagem nova, o Algol 68
- Esta linguagem tem forte base matemática mas é muito difícil de entender e de implementar
- Nenhum compilador completo foi jamais construído mas a linguagem teve influencia significativa em outras linguagens, tais como Pascal, C e Ada.

Pascal

- Criação de Nyklaus Wirth como descendente do Algol
- Iniciou seu desenvolvimento em 1968 e teve uma implementação em 1970
- Compromisso entre as características de uma larga variedade de linguagens com o objetivo de uso por parte de estudantes e eficiência na compilação em um só passo
- Como linguagem de ensino teve tremendo sucesso e é largamente empregada a despeito de suas deficiências para a implementação de sistemas de grande porte.
- Em 1986 foi divulgado o Object Pascal que é a extensão da linguagem Pascal para a orientação a objetos.
- Em 1987 a empresa Borland lançou o ambiente Turbo Pascal 4.0 que foi um grande sucesso pela facilidade de edição, compilação e execução que, pela primeira vez, foram integrados no ambiente de microcomputadores
- O Pascal v7.0 da Borland (para o DOS, Windows 16-bit e a modalidade protegida) foi liberado em 1992 sendo a última versão do Pascal Turbo a ser nomeado desta forma.
- O Pascal 8 Turbo foi chamado Delphi.

Pascal (exemplo)

```
{ Pascal Example Program}
  Input: An integer, listlen, where listlen is less than 100, followed by listlen-integer
         values
  Output: The number of input values that are greater  than the average of the input
         values }
program passex (input, output);
  type intlisttype = array [1..99] of integer;
  var
    intlist: intlisttype;
    listlen, counter, sum, average, result: integer;
  begin
    result:=0;
    sum:=0;
    readln (listlen);
    if ((listlen>0) and (listlen<100)) then
    begin
    for counter:= 1 to listlen do
      begin
        readln (intlist[counter]);
        sum:= sum+ intlist[counter])
      end;
      average := sum div listlen;
      for counter := 1 to listlen do
        if (intlist[counter] > average) then
          result:=result+1;
      writeln ('The number of values that are > average is:',
                result)
    end   { of the then clause of if ((listlen > 0 ...}
    else
    writeln('Error--input listlen is not legal')
  end.
```

C

- Esta linguagem foi implementada por Kernighan e Ritchie, entre 1969 e 1973, nos Laboratórios Bell em paralelo com o desenvolvimento do Sistema Operacional UNIX
- Pode ser usada como linguagem quase funcional e é razoavelmente portável
- Foi largamente distribuída como bônus dos sistema UNIX.

C

- A linguagem C inovou por ser um linguagem de “quase baixo nível”, extremamente compacta e poderosa.
- Introduziu a separação de código em dois tipos de arquivos:
  - Arquivos de cabeçalho (.h) que incluem a declaração de constantes simbólicas, identificadores e protótipos de funções.
  - Arquivos de código (.c) que incluem a definição dos procedimentos e funções (todo o código C é uma função ou composição de funções)

JCL

- Esta linguagem (“Job Control Language”) foi criada para automatizar o OS/360, no início dos anos 70 do século passado
- Foi a primeira linguagem de scripting e, para os padrões atuais, era bastante primitiva
- Sua finalidade era organizar a seqüência de tarefas de um trabalho e os fluxos dos conjuntos de cartões perfurados (ou arquivos correspondentes).

JCL (exemplo)

```
//UTCOBLNK JOB
       SIMOTIME,ACCOUNT,CLASS=1,MSGCLASS=O,NOTIFY=CSIP1
//*
       ************************************************
       ***************
//*              This program is provided by:              *
//*               SimoTime Enterprises, LLC                *
//*        (C) Copyright 1987-2003 All Rights Reserved      *
//*                                                         *
//*          Web Site URL:   http://www.simotime.com        *
//*              e-mail:   helpdesk@simotime.com            *
//*
       ************************************************
       ***************
//*
//* Subject: Compile and Link COBOL program, create a load
       member
//* Author:  SimoTime Enterprises
//* Date:    January 1,1998
//*
//       SET MEM=SIMODUMP
//*
```

```
//COBS1    EXEC COB2UC,
//           PARM='MAP,APOST,DYN,NOOPT,SZ(MAX),
//           NOTERM,DATA(31),LIB,SOURCE'
//SYSLIB   DD  DISP=SHR,DSN=SYS1.COB2LIB
//        DD  DISP=SHR,DSN=SIMOTIME.DEMO.COBCPY1
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD
       DISP=SHR,DSN=SIMOTIME.DEMO.OBJECT(&MEM)
//SYSIN    DD
       DISP=SHR,DSN=SIMOTIME.DEMO.COBOL(&MEM)
//*
//LKED     EXEC
       PGM=IEWL,REGION=2M,PARM=(XREF,LET,LIST,CALL),
//           COND=(8,LT)
//SYSLIB   DD  DISP=SHR,DSN=SYS1.COB2LIB
//        DD  DISP=SHR,DSN=SIMOTIME.DEMO.OBJECT
//SYSLIN   DD
       DISP=SHR,DSN=SIMOTIME.DEMO.OBJECT(&MEM)
//SYSLMOD  DD
       DISP=SHR,DSN=SIMOTIME.DEMO.LOADLIB1(&MEM)
//OBJECT   DD  DISP=SHR,DSN=SIMOTIME.DEMO.OBJECT
//SYSTERM  DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//*
```

REXX

- Criada em 1979 por Michael Cowlishaw, nos laboratórios IBM da Inglaterra, para simplificar as tarefas no ambiente de tempo compartilhado CMS da IBM
- Logo tornou-se uma linguagem de macros para programas de aplicação quaisquer e seu emprego espalhou-se por diversas plataformas incluindo Unix e os PCs
- Sintaxe foi obtida da linguagem PL/I
- Em 1990 foi lançada a versão para Unix e PC
- Nos últimos anos 90 do século XX foi criada a versão Object Rexx
- Em 1996 a versão compatível com Java, NetRexx veio a público.

REXX (exemplo)

```
/* REXX                                   */
 /* Year 2000 CountDown                          */
/* example of date() & time() usage.              */
/* Display current date, plus days, hours, minutes,    */
/* and seconds before year 2000.                  */
/* by: Bruce Gillispie - 03/03/1999 - Tech Support     */
/*                                        */
    'clear'     /* CLS on some systems...             */
 parse value time('N') with hh +2 . +1 mm +2 . +1 ss +2 ;
 parse value date('S') with yyyy +4 mm +2 dd +2 ;
  say 'Today is 'date('W',(date('B')),'B')' 'date('M'),
            strip(dd,,'0')', 'yyyy' (day 'date('D')' of 'yyyy')';
 say 'Only' (date('B',19991231,'S')-date('B')),
            'days' (23-hh) 'hours' (59-mm) 'minutes and',
            (59-ss) 'seconds before the Year 2000.';
exit00
```

PROLOG

- Kowalski, em Edimburgo, formulou uma implementação procedimental da lógica das clausulas de Horn e mostrou que o axioma A if B pode ser lido como um procedimento ou uma linguagem de programação recursiva, aonde A é a cabeça do procedimento e B é o corpo
- Em 1972, Colmeraurer e seu grupo desenvolveram um provador de teoremas que foi usado para a implementação do processamento da linguagem natural
- Este provador de teoremas foi chamado de PROLOG e englobou a interpretação procedimental de Kowalski.

PROLOG (exemplo)

```
mother (joanne, jake).
mother (sue, joanne).
parent (X, Y) :- mother (X,Y).
grandparent(X,Z) :- parent (X,Y), parent (Y,Z).
parent (sue, jill).
grandparent(sue, T).
```

Ada

- Por encomenda do Departamento de Defesa Norte Americano foi criada esta linguagem como descendente direta do Pascal
- Sua especificação final foi de 1981 muito embora os primeiros compiladores já tivessem aparecido em 1980
- È linguagem especial para grandes sistemas e influenciou o desenvolvimento de C++.

Ada (exemplo)

```
With STANDARD_IO;
Use STANDARD_IO;
Procedure HELLO_WORLD is
Begin
   Put("Hello world!");
End HELLO_WORLD;
```

Smalltalk

- Desenvolvida no Palo Alto Research Center, da Xerox tendo sido uma derivação de Simula 67 por Alan Kay em 1971
- Características:
  - Objetos encapsulando métodos e estruturas de dados
  - Objetos definidos como instâncias de classes
  - Classes organizadas em hierarquias de herança
  - Objetos se comunicando por meio de mensagens
- A maioria dos sistemas ditos orientados a objetos possuem as três primeiras características acima
- Muitos omitem a última

C++

- Linguagem desenvolvida nos Bell Labs em 1983 por Bjarn Stroutstrup.
- Descendente da linguagem C com estrutura de classes e algumas restrições
- Características:
  - Classes com herança
  - Possibilidade de múltipla herança
  - Sobrecarga com ligação de tipos dinâmica
  - Chamadas de procedimentos embutidas em vez do modelo de passagem de mensagens, o que é mais eficiente porém menos flexível

.

C++

- Linguagem ainda a mais ajustada para computação de alto desempenho de uso geral, sendo ideal para aplicações cliente-servidor e aplicações que não precisem transitar entre diferentes plataformas
- Nenhuma outra linguagem de uso comum proporciona a mesma potência, grau de controle e desempenho
- Única linguagem moderna, orientada a objetos com as vantagens de linguagem de baixo nível proporcionadas pela linguagem C
- Vem se modernizando tendo sido a primeira a trazer “templates”, herança múltipla e outras abordagens básicas da orientação a objetos para o grande público
- Visual C++ é a implementação de C++ da Microsoft e foi projetada em torno da Microsoft Foundation Class Library (MFC), que esconde a API Windows

C++ (exemplo)

Arquivo shape.h

```
class Shape {

public:
   Shape(int newx, int newy);
   int getX();
   int getY();
   void setX(int newx);
   void setY(int newy);
   void moveTo(int newx, int newy);
   void rMoveTo(int deltax, int deltay);
   virtual void draw();

private:
   int x;
   int y;
};
```

C++ (exemplo)

Arquivo shape.cpp

```
include "Shape.h"

Shape::Shape(int newx, int newy) {
   moveTo(newx, newy);
}

int Shape::getX() { return x; }
int Shape::getY() { return y; }
void Shape::setX(int newx) { x = newx; }
void Shape::setY(int newy) { y = newy; }

void Shape::moveTo(int newx, int newy) {
   setX(newx);
   setY(newy);
}
void Shape::rMoveTo(int deltax, int deltay) {
   moveTo(getX() + deltax, getY() + deltay);
}
void Shape::draw() {
}
```

“Shells” Unix (sh, csh, ksh, ...)

- Os “shells” Unix foram criados no início dos anos 70, sendo empregados para a digitação de comandos interativos e para a escrita de scripts de automação de tarefas comuns
- Um dos aspectos mais poderosos do Unix é a capacidade de elaboração de scripts Shell que criam novas aplicações compondo aplicações existentes
- Talvez tenha sido esta a razão mais poderosa da popularidade dos sistemas Unix.

Exemplo bash (bourne again shell)

```
#!/bin/sh
echo "What is your favourite OS?“
select var in "Linux" "Gnu Hurd" "Free BSD" "Other"; do
        break
Done
echo "You have selected $var“
```

- O que este script faz é o seguinte:

```
 What is your favourite OS?
1)      Linux
2) Gnu Hurd
3) Free BSD4) Other
#? 1
You have selected Linux
```

PERL

- Linguagem criada por Larry Wall em 1986 (“Practical Extraction and Report Language”) como uma linguagem para simples manipulação de textos tornou-se uma linguagem de uso geral, particularmente empregada na Web
- Possui um interpretador que varre todo o código, compila-o e o transforma para um formato interno antes da execução fazendo com que desapareçam os espaços em branco e os comentários
- Código é altamente portátil mas de difícil leitura para quem não está acostumado com a linguagem
- Possui muitas características (“features”) que fazem com que problemas complexos possam ser codificados em poucas linhas. É bastante utilizada em CGI.

PERL (exemplo)

Exemplo de programa:

```
$string1 = "Hello World\n";
if ($string1 =~ m/(H..).(o..)/) {
  print "We matched '$1' and '$2'\n";
}
```

Saída correspondente

```
We matched 'Hel' and 'o W';
```

Tcl

- Linguagem de scripting criada por John Ousterhout em 1989
- A idéia básica constituía em criar uma linguagem de comandos embutidos
- 0bjetivo primordial: criação de interfaces gráficas párea usuários (GUI)
- Conjunto de componentes GUI foi desenvolvido como uma extensão de Tcl e recebeu o nome de Tk
- Tcl/Tk mostrou ser a maneira mais fácil de criar GUI no ambiente Unix
- Alternativas baseadas em C, tais como Motif eram muito mais complicadas e forneciam menor número de funcionalidades gastando entre 5 e 10 vezes mais esforço do que a alternativa Tcl/Tk
- Tcl é facilmente embutida em qualquer código
- Em 1994 Ousterhout mudou-se para a Sun e aumentou a capacidade de recursos pois o desenvolvimento inicial foi em ambiente acadêmico (Berkeley)
- Possui bastante penetração nos ambientes Unix e McIntosh.

Tcl (exemplo)

- Primeiro exemplo

```
set x [new HelloWorld]
puts “[$x hello] [$x World]”
delete $x
```

- Segundo exemplo

```
set $x [new fraction 1 3]
set $y [new fraction 2 9]
set $z [new fraction 3 27]

$x += $y
$x /= $z

$x value
$x numden

delete $x $y $z
```

Python

- Python é uma linguagem imperativa, interpretada, interativa comparada a Perl e Java
- Considerada por seu autor (Guido van Rossum – Holanda 1987) como uma linguagem de scripting
- Ideal para desenvolvimento de protótipos
- Coleta de lixo automática para gerenciamento de memória
- Bastante adequada para escrever pequenos trechos de código
- Pode receber módulos escritos em linguagens compiladas, tais como C e C++
- Interpretador pode ser embutido em uma aplicação para garantir uma interface programável
- A idéia inicial era preencher o vazio entre os “shells” Unix e a programação C.

Python (exemplo)

- Exemplo de um função simples em Python que inverte uma tabela representada como um dicionário Python.

```
def invert(table):
    inverted = {}              # empty dictionary
    for key in table:
        value = table[key]
        # Each key must be unique, but there could be duplicate values
        if not index.has_key(value):
            inverted[value] = [] # empty list
        inverted[value].append(key)
    return inverted
```

Python (exemplo)

- Pode-se notar que Python usa a endentação para agrupamento de comandos. Os comentários são precedidos por um caractere #. O exemplo seguinte é de uma função interativa. O “prompt” do interpretador é ">>> ".

```
 >>> phonebook = {'guido': 4127, 'sjoerd': 4127, 'jack': 4098}
>>> phonebook['dcab'] = 4147          # add an entry
>>> inverted_phonebook = invert(phonebook)
>>> print inverted_phonebook
{4098: ['jack'],
4127: ['guido', 'sjoerd'],
 4147: ['dcab']}
>>>
```

WFL

- “Work Flow Language” ou WFL foi a linguagem de controle da Burroughs
- Era muito mais poderosa do que a concorrente JCL tendo a sintaxe da linguagem ALGOL e servia para organizar a seqüência de tarefas de um trabalho e definir o ambiente nos sistema Burroughs.

C#

- C# é uma linguagem Microsoft que usa uma sintaxe simplificada, semelhante à da linguagem C e projetada especificamente para uso na Web. Foi desenvolvida em 1999.
- O .NET Framework é escrito em C#, linguagem esta que foi projetada para ser de uso mais fácil e de maior produtividade que C++. O código é mais legível, não se usam arquivos de cabeçalhos (.h), as operações ""“>""e “::” são substituídas por “.”
- O comando switch não precisa de um “break” em cada cláusula e os inteiros não sofrem conversão forçada para booleano nunca.
- Não aceita herança múltipla, só permitindo às classe implementarem múltiplas interfaces.
- Não suporta ponteiros.

C#

- Ao contrário de linguagens tais como C e C++, C # não compila para código binário, que pode ser executado diretamente pelo computador alvo
- Em vez disto todas as linguagens do NET (que inclui o NET Visual Basic e C++ , além de C #) compilam para um alvo que a Microsoft chama de MSIL
- Ao observador ocasional, o programa resultante parece com um executável normal e tem uma extensão de ".exe" exatamente como uma aplicação normal
- Entretanto o programa do NET consiste realmente em uma linguagem intermediária

C#

- Quando o programa for executado, as NET framework fazem uma compilação JIT (Just-In-Time tal como a Java Virtual Machine de Java) com o código intermediário para a linguagem de máquina à medida que roda
- A compilação JIT é muito rápida e não é sequer perceptível na maioria de PC modernos
- Quando as partes diferentes do programa são usadas, elas são traduzidas
- Uma vez traduzida, uma parcela do programa não necessita ser traduzida outra vez para a mesma execução do programa (o programa usará a versão traduzida)
- Cada vez que uma aplicação do NET é executada, necessita ser traduzida desta forma
- Já que as aplicações do NET requerem esta tradução, apenas os computadores com a estrutura do NET podem funcionar aplicações do NET.

C# (exemplo)

```
using System;
public class Hello
{
public static void Main()
{
Console.WriteLine("Hello please work.. please please
please..");
}
}
```

Visual Basic (VB)

- Em março de 1988 Alan Cooper desenvolveu o produto “drag-and-drop” Tripod e negociou o conceito deste produto, cognominado Ruby com a Microsoft
- Bill Gates comprou a empresa.
- Em 1991 a PowerSoft lançou o PowerBuilder
- Em março de 1991 foi lançado o VB1
- Em novembro de 1992 foi lançado o VB2 com suporte a ODBC. Nessa ocasião foi lançado o Access.
- Em junho de 1993 foi lançado o VB3 e o revolucionário Data Control
- Em maio de 1995 foi lançado o Delphi 1.0
- No segundo semestre de 1996 foi lançado o IE 3.0
- VB4 foi lançado em outubro de 1996
- VB5 foi lançado em abril de 1997
- Em outubro de 1998 foi lançado VB6.
- Em 2001 foi lançado o Visual Basic .NET

Delphi

- A Borland pretendeu lançar um produto “VB-killer”
- Delphi teve seu nome originado da capacidade desejada de acesso a Bancos de Dados tipo Oracle
- Ocorre que Oracle foi derivado da entidade mitológica que sabia tudo o Oráculo de Delfos ou “Delphi Oracle”
- Um dos nomes cogitados foi AppBuilder mas, enquanto se desenvolvia o produto a Novell lançou o Visual AppBuilder

Delphi

Versão	Data	Característica
1	1995	Identificadores com até 255 caracteres, alternativa ao VB
2	1996	Versão para 32 bits; melhor acesso a BD
3	1997	Versão muito estável; herança múltipla; Variants
4	1998	Versão com problemas; “arrays” dinâmicos; assert
5	1999	Sobrecarga de métodos; Kylix
6	2001	Ferramentas XML
7	2002	Plataforma .NET; integração com UML

Delphi

- A biblioteca de componentes visuais (VCL) é uma hierarquia das classes -- escritas em Object Pascal e amarradas ao Delphi IDE (Integrated Development Environment) -- que permite que você desenvolva aplicações rapidamente.
- Por definição todos os objetos descendem de TObject
- Todos os objetos de Delphi que você pode utilizar em um formulário descendem de TComponent, e qualquer coisa que envolva um controle de janelas descende de TWinControl
- Apesar de seu nome, o VCL consiste na maior parte em objetos não visuais.

Java

- A linguagem Java é um produto da empresa Sun Microsystems - comprada pela Oracle
- O objetivo era projetar uma tecnologia que poderia integrar dispositivos eletrônicos do consumidor usando uma linguagem de programação padrão através da Internet
- Pela complexidade e “não confiabilidade” da linguagem C++, James Gosling preparou-se em junho de 1991, um interpretador "Oak" -- orientado a objetos (OO)
- Hoje, Gosling é considerado o "pai da linguagem Java“
- Originalmente, o compilador Java foi escrito em C
- Como prova da viabilidade da linguagem Java, o compilador foi reescrito na linguagem Java em 1994.

Java

- Em maio de 1995, a Sun anunciou a linguagem Java
- Logo todos os fabricantes principais de software licenciaram a tecnologia de Java
- O primeiro uso de Java na Internet eram as applets (aplicações Java que devem rodar sob o controle de um navegador dentro de uma página Web HTML)
- Web sites adicionaram applets para executar os efeitos especiais de texto ou de gráficos, que deram outra vida às enfadonhas páginas estáticas
- Logo surgiram as aplicações Java autônomas
- O uso o mais difundido da tecnologia Java está em aplicações do servidor, usando servlets , JavaServer Pages (JSP) e Enterprise JavaBeans (EJBs).

Java (exemplo)

```
import java.sql.*;
import corejava.*;
public class Teste
  {public static void main (String[] args)
    {Connection conn = null;

try
  {Class.forName("oracle.jdbc.driver.OracleDriver");
    conn = DriverManager.getConnection
       ("jdbc:oracle:thin:@sbd1:1521:orcl","java01","java01");
    conn.setAutoCommit(false);
  }
catch (Exception e)
  {System.out.println ('\n' + "Erro na conexão com o banco.");
   e.printStackTrace();
   System.exit(1);
   }

String nome = "AAA";
String cpf = "123";
double sal = 2000;
String umString;
umString = Console.readLine('\n' +
 "Inserindo linhas com Driver Puro Java. Aperte uma tecla para continuar...");
```

Java (exemplo)

```
    try
       {PreparedStatement stm = conn.prepareStatement
        ("INSERT INTO EMPREGADOS (NUMERO, NOME, CPF, SALARIO) " +
        "VALUES (EMPREGADOS_SEQ.NEXTVAL, ?, ?, ?)");
        for (int i = 1; i < 10; i++)
          {sal = sal + 1;
           stm.setString(1, nome);
           stm.setString(2, cpf);
           stm.setDouble(3, sal);
           stm.executeUpdate();
           conn.commit();
          }
            stm.close();
         }
     catch (java.sql.SQLException e)
         {e.printStackTrace();
         System.exit(1);
        }
    }
}
```

C++ Builder

- Em janeiro de 1997 foi lançado o C++ Builder como sendo uma “versão Delphi C++”, ou seja, aonde o Delphi era constituído de IDE, VCL e Object Pascal, o C++ Builder é constituído de IDE, VCL e C++.