

Pascal

**Caio César, Diego Feitosa, Felipe
Guerzoni, Guilherme Silva e
Mateus Diniz**



Sumário

1 Introdução

2 Histórico

3 Paradigma

4 Características

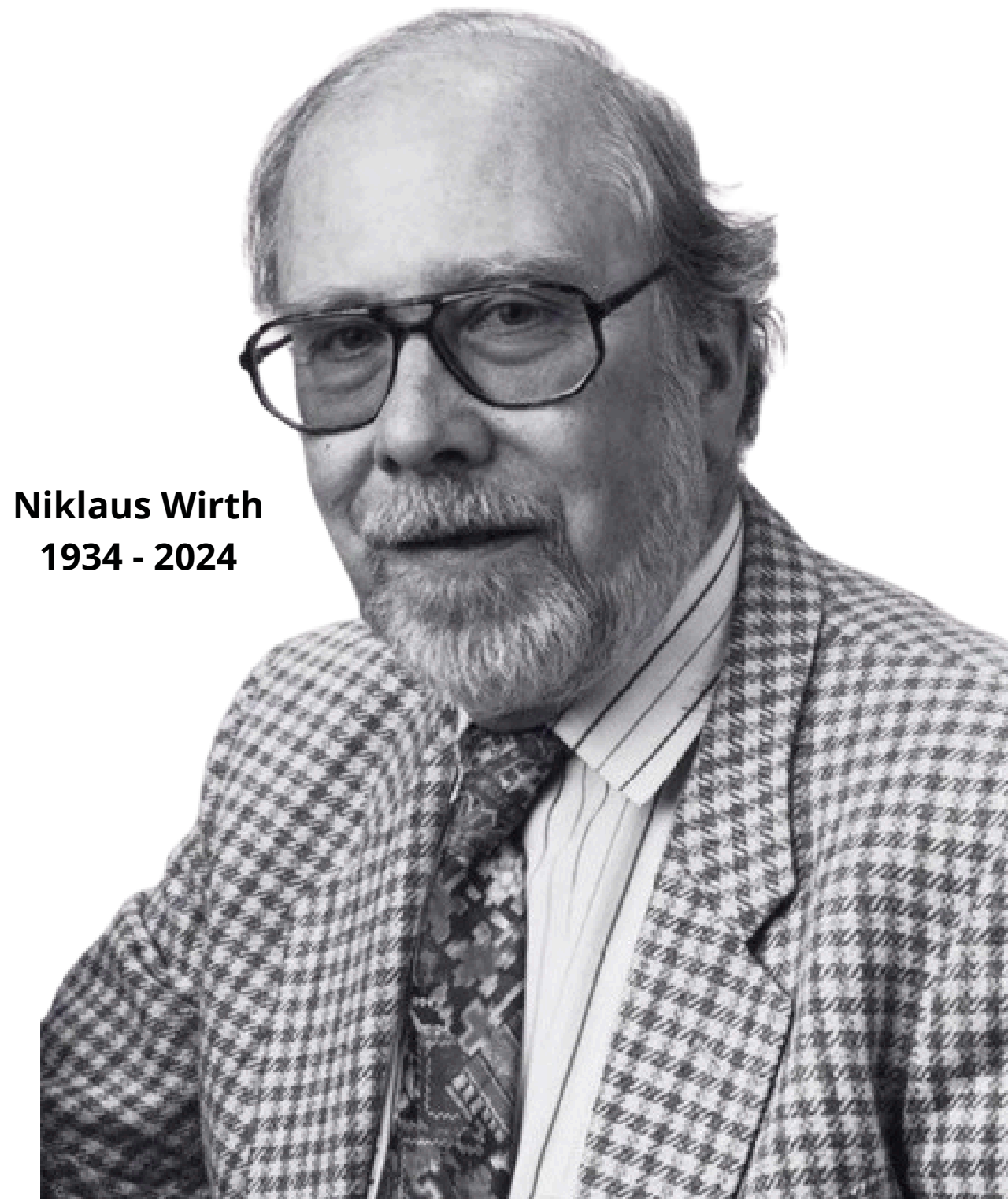
5 Linguagens relacionadas

6 Exemplos práticos

7 Considerações finais

8 Bibliografia

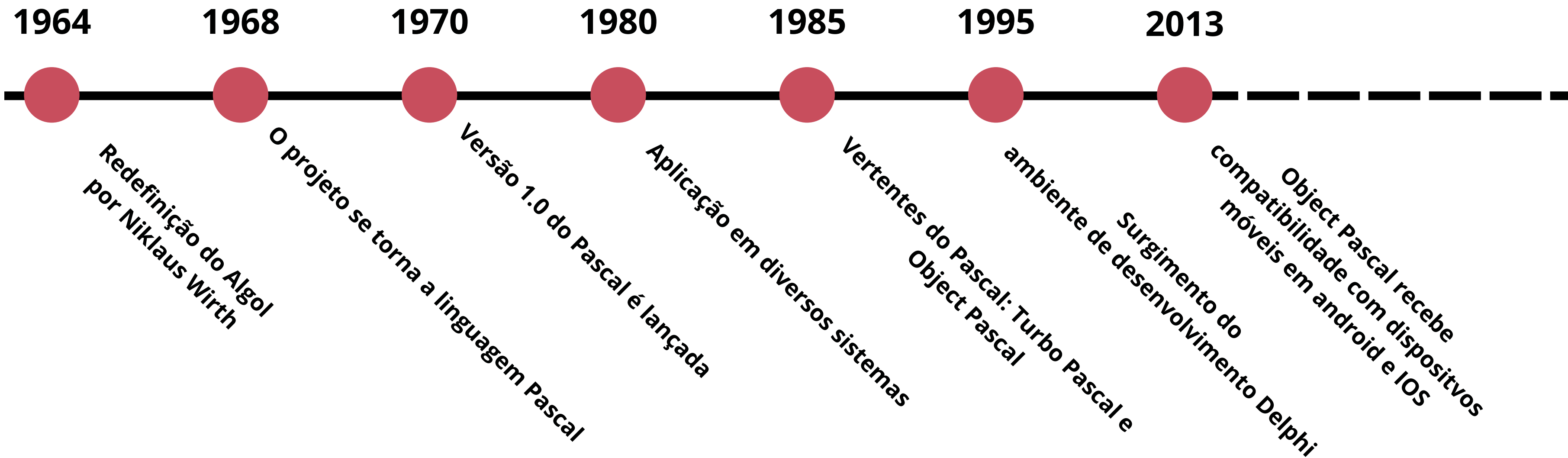
Introdução



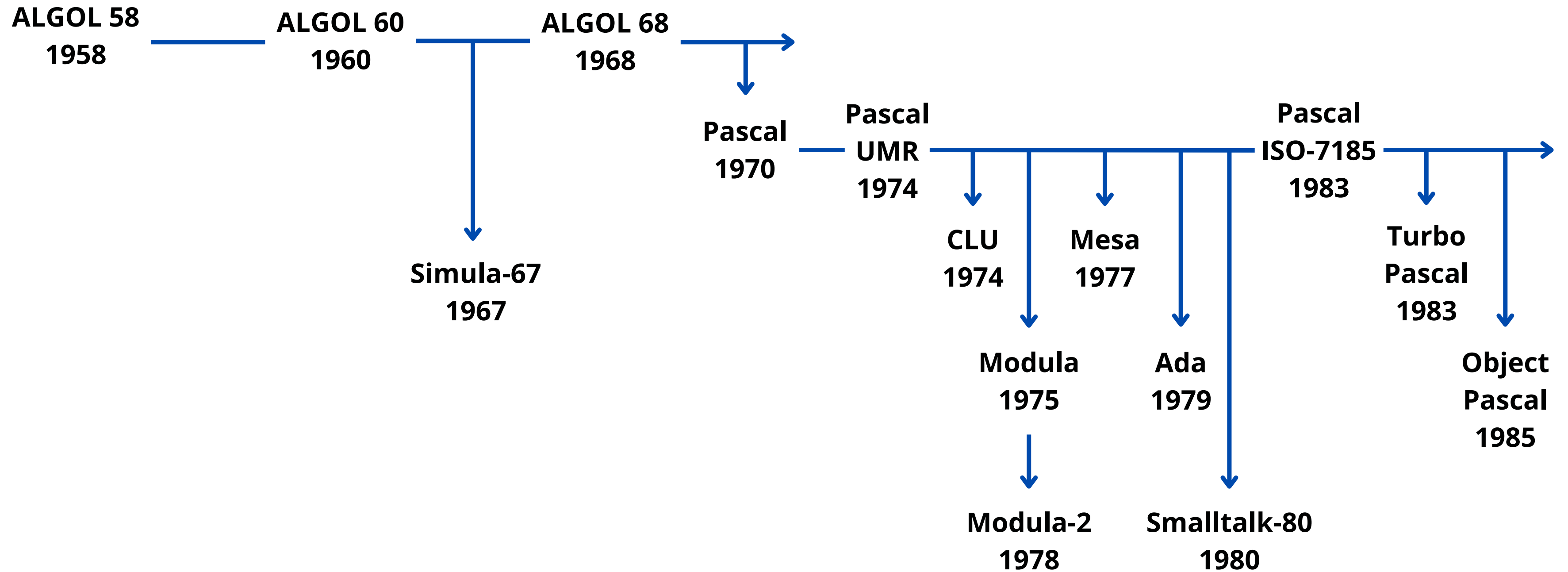
Niklaus Wirth
1934 - 2024

- Desenvolvida no final da década de 1960.
- Linguagem de tipagem forte e sintaxe simples.
- Uma das principais concorrentes da linguagem C por anos.

Histórico



Histórico



Paradigm

Imperativo

Execução de comandos que atualizam variáveis, com a presença de:

- Variáveis
- Comandos
- Procedimentos

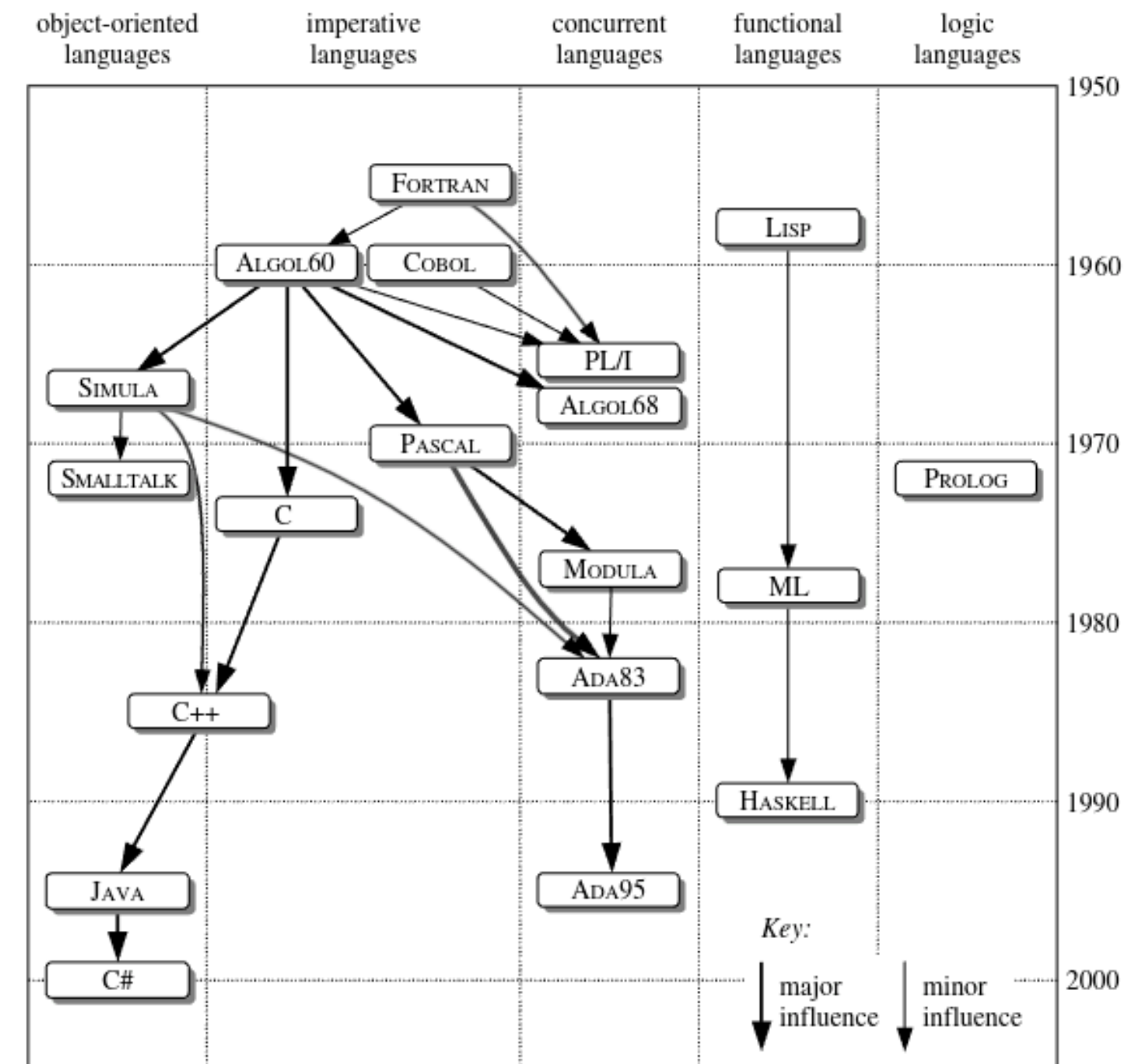


Figure 1.1 Dates and ancestry of major programming languages.

Características

Linguagem educacional

- Foco no ensino
- Incentivo a boas práticas
- Linguagem clara e objetiva

Características

Tipagem estática

```
var
  x: Integer; y: Real;
begin
  x := 10; x := y;
end.
```

```
var
  x: Integer; y: Real;
begin
  x := 10; x := Trunc(y);
end.
```


Características

Sintaxe

```
function factorial(const a: integer) : integer;  
begin  
    if a ≥ 1 then  
        factorial := a * factorial(a-1)  
    else  
        factorial := 1;  
    end;  
end;
```

```
procedure factorial(const a: integer; var result: integer);  
begin  
    if a ≥ 1 then  
        begin  
            factorial(a - 1, result);  
            result := a * result;  
        end  
    else  
        result := 1;  
    end;  
end;
```

Características

Linguagem educacional

- Foco no ensino
- Incentivo a boas práticas
- Linguagem clara e objetiva

Tipagem estática

```
var
  x: Integer; y: Real;
begin
  x := 10; x := y;
end.
```

```
var
  x: Integer; y: Real;
begin
  x := 10; x := Trunc(y);
end.
```

Sintaxe

```
function factorial(const a: integer) : integer;
begin
  if a ≥ 1 then
    factorial := a * factorial(a-1)
  else
    factorial := 1;
end;
```

```
procedure factorial(const a: integer; var result: integer);
begin
  if a ≥ 1 then
    begin
      factorial(a - 1, result);
      result := a * result;
    end
  else
    result := 1;
end;
```

Linguagens Relacionadas



Uma linguagem baseada em pascal criada pelo departamento de defesa americano com o objetivo de possuir alta segurança e confiabilidade para sistemas críticos como aeronaves

Algol-68

Sendo o precursor para muitas linguagens como C, Python e Pascal, Algol 68 trouxe conceitos avançados que revolucionaram a indústria na época como o uso de estruturas dinâmicas e tipagem forte, evitando a ocorrência de erros

Modula-2

Estabelecida pelo mesmo desenvolvedor do pascal, modula foi criada com o propósito de ser mais modular e segura, além de possuir compatibilidade com sistemas embarcados

Pascal

```
program PrimoCheck;
uses crt;

function EhPrimo(n: Integer): Boolean;
var
  i: Integer;
begin
  if (n < 2) then
    EhPrimo := False
  else
    begin
      EhPrimo := True;
      for i := 2 to n div 2 do
        begin
          if (n mod i = 0) then
            begin
              EhPrimo := False;
              Break;
            end;
        end;
      end;
    end;
end;

var
  num: Integer;
begin
  clrscr;
  Write('Digite um número: ');
  ReadLn(num);

  if EhPrimo(num) then
    WriteLn('O número ', num, ' é primo.')
  else
    WriteLn('O número ', num, ' não é primo.');
```

```
    ReadLn;
end.
```

Oberon

```
MODULE PrimoCheck;
IMPORT In, Out;

PROCEDURE EhPrimo(n: INTEGER): BOOLEAN;
VAR i: INTEGER;
BEGIN
  IF n < 2 THEN RETURN FALSE END;
  FOR i := 2 TO n DIV 2 DO
    IF (n MOD i = 0) THEN RETURN FALSE END;
  END;
  RETURN TRUE;
END EhPrimo;

VAR num: INTEGER;
BEGIN
  Out.String("Digite um número: ");
  In.Int(num);

  IF EhPrimo(num) THEN
    Out.String("O número "); Out.Int(num, 0); Out.String(" é primo.");
  ELSE
    Out.String("O número "); Out.Int(num, 0); Out.String(" não é primo.");
  END;
  Out.Ln;
END PrimoCheck.
```

Pascal

```
program Fibonacci;
uses crt;

function Fibonacci(n: integer): integer;
    if (n = 0) then
        Fibonacci := 0
    else if (n = 1) then
        Fibonacci := 1
    else
        Fibonacci := Fibonacci(n - 1) + Fibonacci(n - 2); //
end;

var
    i, num: integer;
begin
    clrscr;
    write('Digite um número: ');
    readln(num);
    writeln('Sequência de Fibonacci até ', num, ' termos:');
    for i := 0 to num - 1 do
        write(Fibonacci(i), ' ');
    writeln;
    readln;
end.
```

Haskell

```
fibonacci :: Int -> Int
fibonacci 0 = 0
fibonacci 1 = 1
fibonacci n = fibonacci (n - 1) + fibonacci (n - 2)

main :: IO ()
main = do
    putStrLn "Digite um número: "
    input <- getLine
    let num = read input :: Int
    putStrLn $ "Sequência de Fibonacci até " ++ show num ++ " termos:"
    print [fibonacci n | n <- [0..(num-1)]]
```

Exemplos práticos



Considerações finais

- **Origem e evolução**
- **Características e paradigmas**
- **Comparativo com outras linguagens**
- **Influência**
- **Educação e formação**

Bibliografia

Free Pascal - Advanced open source Pascal compiler for Pascal and Object Pascal - Home Page.
Disponível em: <<https://www.freepascal.org/>>.

MATTHIAS. Free Pascal meets SDL. Disponível em: <<https://www.freepascal-meets-sdl.net/>>.

WATT, D. A.; FINDLAY, W. Programming Language Design Concepts. [s.l.] John Wiley & Sons, 2004.

Pascal - Quick Guide. Disponível em:
<https://www.tutorialspoint.com/pascal/pascal_quick_guide.htm>.

REISER, M.; WIRTH, N. Programming in Oberon. [s.l.] Addison-Wesley Longman, 1992.

JENSEN, K.; WIRTH, N. PASCAL User Manual and Report. [s.l.] Springer, 2013.

A Gentle Introduction to Haskell, Version 98. Disponível em: <<https://www.haskell.org/tutorial/>>.

Code Examples. Disponível em: <<https://oberon-lang.github.io/>>.

Obrigado.



PUC Minas