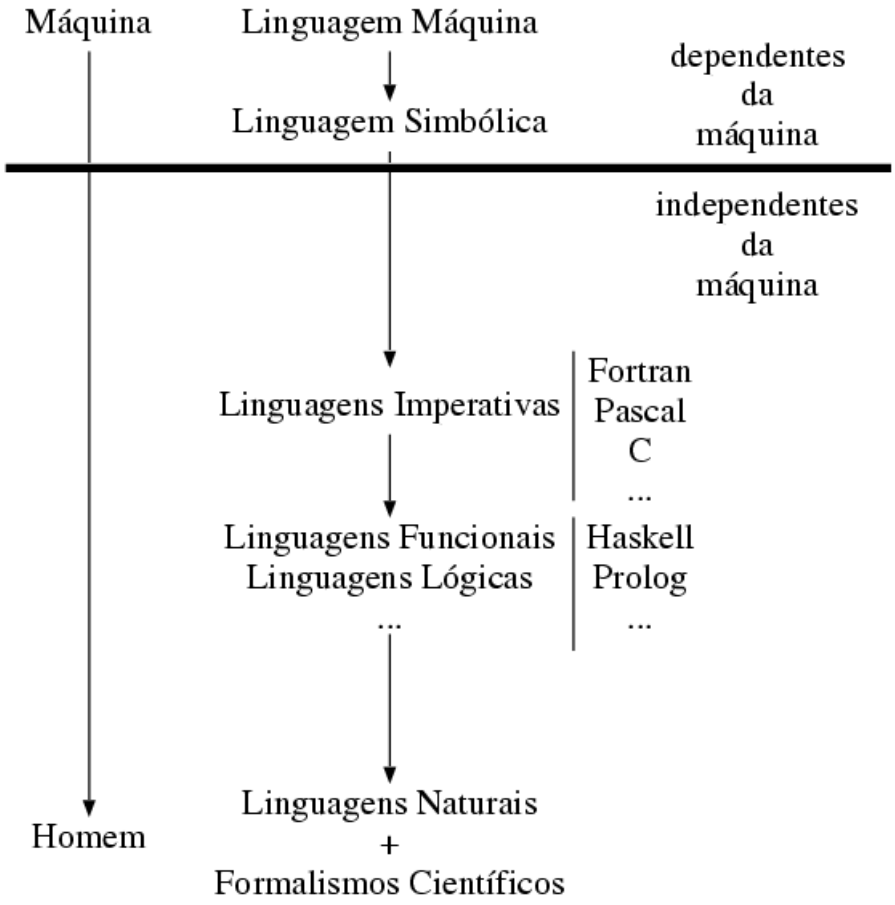
# Linguagens de Programação:

Como comunicar com o computador?




# Linguagem Máquina:

Conjunto básico de instruções, em código binário, características de cada computador, correspondentes às suas **Operações Básicas:**

- Instruções de Cálculo (and, or, not, +, –, …)
- Inst. de Transferência de Informação
- Inst. de Teste
- Inst. de Entrada/Saída

## Linguagem Simbólica (Assembly):

Conjunto de mnemónicas das instruções em código máquina. A tradução é feita pelo programa "Assembler".

Como há uma correspondência biunívoca entre instruções simbólicas e instruções máquina, as linguagens simbólicas:

- Dependem do processador utilizado,
- Permitem escrever programas muito eficientes,
- São de utilização muito difícil e sujeita a erros.

## Linguagens de Alto Nível:

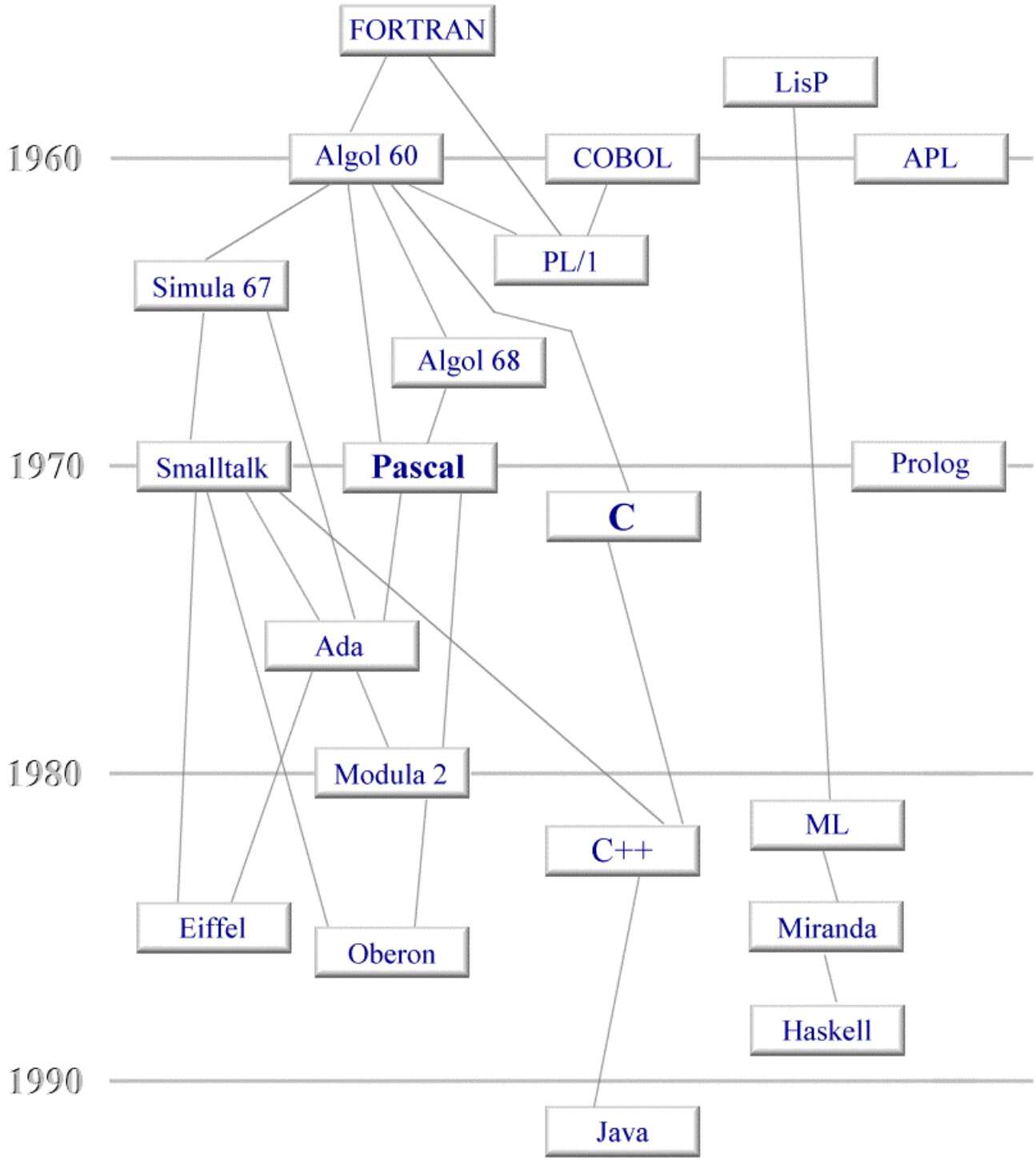
Exemplo em Pascal:

```
begin   pi:= 3.14159;
        writeln ('Escreva o valor do raio');
        read(raio);
        area:= pi * raio * raio;
        comp:= 2 * pi * raio;
        writeln ('Area do circulo =', area);
        writeln ('Comprimento da circunferencia =', comp)
end.
```

As Linguagens de Alto Nível são:

- Mais próximas dos conceitos humanos (linguagem natural, conceitos matemáticos),
- Independentes do computador,
- Cada instrução corresponde a uma lista de instruções em linguagem máquina (ou simbólica),
- Permitem uma programação mais fácil e menos sujeita a erros,
- Os programas são mais curtos e fáceis de ler, entender e alterar,
- Não permitem o controlo exacto da máquina.

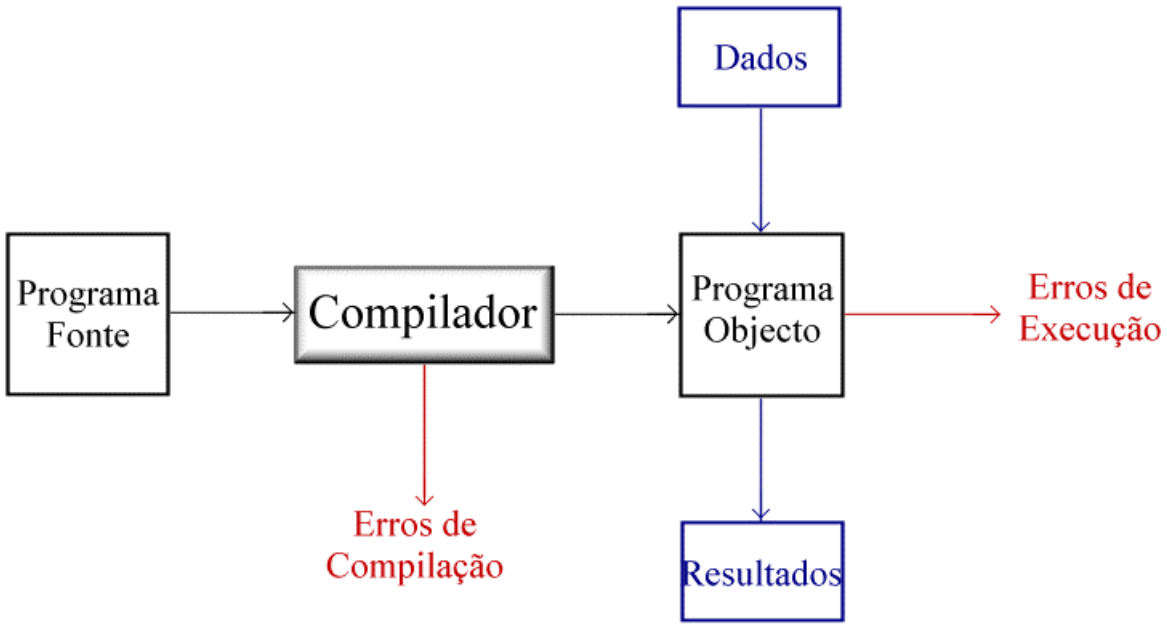
# Cronologia de algumas Linguagens de Alto Nível:



# O programa Compilador:

Traduz um dado programa em linguagem de Alto Nível (**Programa Fonte**) para o seu equivalente em Linguagem Máquina **(Programa Objecto**).

A **Fase de Compilação** gera código executável pelo computador onde, durante a **Fase de Execução**, podemos introduzir os **Dados** e esperar os **Resultados** correctos.

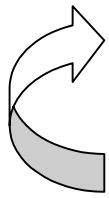



Erros de Compilação: O Programa Fonte não foi escrito de acordo com as regras de Sintaxe da linguagem utilizada.
O Compilador não gera o Programa Objecto.

Erros de Execução: O Programa Objecto não pode ser executado.

Erros do Algoritmo: Os resultados foram calculados, mas estão errados.

**Programar ≡ Resolver Problemas**



Problema → Especificação → Elaboração do Algoritmo → Construção das Estruturas de Dados → Escrita do Programa → Implementação → Compilação → Execução → Análise dos Resultados → Correcção dos Erros

## 1. Problema:

Calcular em que dia da semana calha o dia de Natal de ano qualquer.
p.ex.: Natal (1985) = Quarta-Feira
Natal (1999) = Sábado

## 2. Especificação:

**Dados:** Indicação do **ano** em questão.
(Definição do domínio?)

**Resultados:** Identificação de um dos sete dias da semana.
(Como? Codificação?)

# 3. Elaboração do Algoritmo:

## Relações entre os Dados e os Resultados:

Não parece haver uma relação matemática directa …

## Análise de alguns casos particulares:

Natal (1997) = Quinta
Natal (1998) = Sexta
Natal (1999) = Sábado
Natal (2000) = Segunda **porquê o "salto"?**

## Algumas informações úteis:

Um ano comum tem 365 dias, um ano bissexto tem 366;
O dia extra dos anos bissextos é o 29 de Fevereiro;
O Natal é a 25 de Dezembro;
O ano 2000 é bissexto.

**Será que?:** Natal(ano+1) = Natal(ano) + **1**, se (ano+1) fôr comum
Natal(ano+1) = Natal(ano) + **2**, se (ano+1) fôr bissexto

## Porquê?

365 mod 7 = **1** ou 365 = 7 × 52 + **1**
366 mod 7 = **2** ou 366 = 7 × 52 + **2**

## Mas quais são os anos bissextos?

**Um ano é bissexto se fôr divisível por 4,
mas não por 100, ou se fôr divisível por 400.**

Exemplos: 1984, 1988, 1992, 1996 são anos bissextos,
1900, 2100 são anos comuns,
2000 é ano bissexto.

## Uma representação numérica para os dias da semana:

domingo ↔ 0
segunda ↔ 1
…
sabado ↔ 6

### Verifiquemos…

Natal (1998) = 5 (sexta)
Natal (1999) = 5 + 1 = 6 (sábado)
Natal (2000) = 6 + 2 = **8** … e deveria ser 1

## O resultado tem de ser reduzido ao intervalo inteiro [0 , 6]

### Alteremos as fórmulas:

Natal(ano+1) = (Natal(ano) + 1) mod 7, se (ano+1) fôr comum
Natal(ano+1) = (Natal(ano) + 2) mod 7, se (ano+1) fôr bissexto

### Serão necessárias duas fórmulas?

Natal(ano+1) = (Natal(ano) + 1 + **bis**) mod 7

com **bis** = **0** se (ano+1) comum
**bis** = **1** se (ano+1) bissexto

## Generalizando para n anos depois:

Natal(ano + n) = (Natal(ano) + n + **numbis**) mod 7

com **numbis** = número de anos bissextos durante esses n anos

## Quantos anos bissextos ocorrem no intervalo [ano , ano+n]?


(ano)

Se **ano** fôr bissexto
Então **numbis** ([ano , ano+n]) = **n div 4** (divisão inteira)
(desde que o intervalo não contenha anos divisíveis por 100)

## Escolha do domínio:

Como o Natal é em Dezembro, podemos escolher 1900 como **ano** base e "permitir" o cálculo pretendido no intervalo [1900 , 2099]

## E só falta descobrir que:

**Natal(1900) = terça-feira (2)**

## Finalmente, o Algoritmo procurado:

**Cálculo do dia de Natal:**

Ler o valor do ano pretendido ∈ [1900 , 2099];
Calcular n ← ano – 1900;
Calcular numbis ← n div 4;
Calcular factor ← 2 + n + numbis;
Calcular natal ← factor mod 7;
Escrever o valor de natal (dia da semana).

## 4. Simulação para alguns valores:

| ano | n | numbis | factor | natal | dia |
|---|---|---|---|---|---|
| 1900 | 0 | 0 | 2 | 2 | terça |
| 1999 | 99 | 24 | 125 | 6 | sábado |
| 2000 | 100 | 25 | 127 | 1 | segunda |

… parece funcionar!

## 5. Uma implementação na linguagem Pascal:

```
(* Calcular o dia da semana em que calha o Natal *)
(* Utilizar só para anos entre 1900 e 2099 *)
program diadenatal(input, output);
var ano, numbis, n, factor, natal: integer;

begin   write('Qual o ano, só dos séculos XX e XXI?');
        readln(ano);
        n:=ano −1900;
        numbis:=n div 4;
        factor:=2+n+numbis;
        natal:=factor mod 7;
        write('O Dia de Natal de ', ano,' calha a ');
        case  natal of
                0 : writeln('um Domingo');
                1 : writeln('uma Segunda-Feira');
                2 : writeln('uma Terca-Feira');
                3 : writeln('uma Quarta-Feira');
                4 : writeln('uma Quinta-Feira');
                5 : writeln('uma Sexta-Feira');
                6 : writeln('um Sabado')
        end
end.
```

## 6. Editar 7. Compilar 8. Executar
## 9. Corrigir Erros (… e voltar ao início?)

## 10. Aperfeiçoamentos:

Melhoramentos e adaptações possíveis:

- Reduzir o número de variáveis utilizadas;
- Introduzir mais comentários;
- Ampliar o domínio [1900 , 2099];
- Permitir o cálculo para vários anos;
- Gerar uma tabela, para um intervalo pedido;
- …

## Metodologia da Programação:

Objectivos a atingir { **Clareza** **Correcção** **Eficiência**

**Clareza:** O programa deve reflectir claramente a estrutura do algoritmo. Deve ser fácil de **ler**, corrigir, ampliar ou modificar, mesmo por outro programador.

**Correcção:** O programa deve cumprir exactamente as especificações.

**Eficiência:** O programa deve tentar minimizar, tanto o seu **tempo** de execução, como o **espaço** de memória utilizado.