

Fundamentos de Programação em Pascal

**Do algoritmo ao programa — exemplo integrador: Jogo da Velha
(texto)**

Autor: (preencha)

Disciplina: Linguagem de Programação I

Objetivos Gerais

- Compreender o que é programar
- Diferenciar algoritmo de programa
- Conhecer a estrutura básica de um programa Pascal
- Explorar tipos, variáveis, operadores, entrada e saída
- Aplicar decisão e repetição
- Consolidar tudo em um pequeno exemplo integrador

O Que é Programar?

Programar = transformar uma ideia (solução) em uma sequência clara de instruções que o computador executa.

Essência: lógica + precisão + clareza.

Algoritmo — Definição

Sequência finita e ordenada de passos para resolver um problema. Características:

- Determinístico
- Geral (independente de linguagem)
- Finito

Representações: texto, pseudocódigo, fluxograma, tabela de decisão.

Exemplo de Algoritmo (Pseudocódigo) — Média de 3 Notas

```
ler n1, n2, n3
media <- (n1 + n2 + n3) / 3
se media >= 7 então
    escrever "Aprovado"
senão
    escrever "Reprovado"
fim-se
```

Do Algoritmo ao Programa

1. Especificar problema
2. Elaborar algoritmo
3. Escolher linguagem (Pascal)
4. Codificar (tradução para sintaxe Pascal)
5. Compilar / Executar / Testar

Estrutura Básica de um Programa Pascal

```
program NomeDoPrograma;  
  
var  
  x, y: integer;  
  
begin  
  x := 10;  
  y := x * 2;  
  writeln('Resultado: ', y);  
end.
```

Blocos principais: cabeçalho (program), declaração de variáveis (var), corpo (begin ... end.).

Tipos Primitivos

Tipo	Uso	Exemplo
integer	inteiros	5
real	números com fração	3.14
char	caractere único	'A'
boolean	lógico (true/false)	true
string	sequência de caracteres	'Teste'

Variáveis

Armazenam valores temporários.

Declaração: nome: tipo;

Inicialização explícita é boa prática.

Exemplo:

```
var  
    nota1, nota2: real;  
    nome: string;  
    aprovado: boolean;
```

Atribuição e Operadores

Atribuição: variavel := expressão;

Operadores aritméticos: + - * / div mod

Operadores relacionais: = <> > < >= <=

Operadores lógicos: and or not

Exemplo:

```
soma := a + b;  
media := soma / 2;  
aprovado := media >= 7; // boolean
```

Entrada e Saída

Saída: `write` , `writeln`

Entrada: `read` , `readln`

```
write('Digite a idade: ');
readln(idade);
writeln('Você digitou: ', idade);
```

`readln` lê e consome a linha; `read` deixa o ENTER pendente.

Decisão (if / else)

```
if media >= 7 then
    writeln('Aprovado')
else
    writeln('Reprovado');
```

Aninhamento possível, mas cuidado com legibilidade.

Decisão (case)

```
case opcao of
  1: writeln('Cadastrar');
  2: writeln('Listar');
  3: writeln('Sair');
else
  writeln('Opção inválida');
end;
```

Bom para múltiplas alternativas simples.

Repetição (while)

```
while contador < 5 do
begin
  writeln('Valor: ', contador);
  contador := contador + 1;
end;
```

Executa enquanto a condição permanece verdadeira.

Repetição (repeat ... until)

```
repeat
    readln(valor);
until valor >= 0;
```

Executa pelo menos uma vez. Sai quando condição se torna verdadeira.

Repetição (for)

```
for i := 1 to 10 do  
    writeln(i);
```

Quando sabemos previamente o intervalo.

Boas Práticas Iniciais

- Indentação consistente
- Nomes significativos (`notaFinal` , não `nf1`)
- Comentários curtos e claros
- Evitar código duplicado (mais tarde: funções / procedimentos / vetores)

Erros Comuns

- Esquecer `;`
- Trocar `=` (comparação) por `:=` (atribuição)
- Usar variável não inicializada
- Loop infinito (condição nunca muda)

Diagnóstico: ler mensagem do compilador e localizar linha mencionada.

Mini-Exemplo 1 — Média Simples

```
program MediaSimples;
var n1, n2, media: real;
begin
  readln(n1); readln(n2);
  media := (n1 + n2)/2;
  if media >= 7 then writeln('Aprovado') else writeln('Reprovado');
end.
```

Mini-Exemplo 2 — Conversão Celsius → Fahrenheit

```
program Conversao;
var c, f: real;
begin
  readln(c);
  f := (c * 9/5) + 32;
  writeln('Fahrenheit: ', f:0:2);
end.
```

f:0:2 formata com 2 casas decimais.

Mini-Exemplo 3 — Contador

```
program Contador;
var i: integer;
begin
  i := 0;
  while i < 5 do
    begin
      writeln('i = ', i);
      i := i + 1;
    end;
end.
```

Exemplo Integrador — Jogo da Velha (Visão de Algoritmo)

1. Criar 9 posições vazias
2. Jogador inicial = 'X'
3. Enquanto jogo não terminar:
 - Mostrar tabuleiro
 - Ler escolha (1..9)
 - Verificar se posição livre
 - Marcar
 - Atualizar jogadas
 - Testar vitória
 - Testar empate (9 jogadas)
 - Alternar jogador
4. Exibir resultado final

Jogo da Velha — Estrutura de Dados (Sem Matrizes)

Cada casa é uma variável `char : pos1` a `pos9`.

Controle de estado: `jogador` , `jogadas` , `vitoria` , `fim`.

Jogo da Velha — Declaração e Inicialização

```
var
  pos1,pos2,pos3: char;
  pos4,pos5,pos6: char;
  pos7,pos8,pos9: char;
  jogador: char;
  jogadas, escolha: integer;
  vitoria, fim: boolean;

  pos1 := ' '; pos2 := ' '; pos3 := ' ';
  pos4 := ' '; pos5 := ' '; pos6 := ' ';
  pos7 := ' '; pos8 := ' '; pos9 := ' ';

  jogador := 'X'; jogadas := 0; vitoria := false; fim := false;
```

Jogo da Velha — Mostrar Tabuleiro

```
writeln(' ', pos1, ' | ', pos2, ' | ', pos3);
writeln('-----');
writeln(' ', pos4, ' | ', pos5, ' | ', pos6);
writeln('-----');
writeln(' ', pos7, ' | ', pos8, ' | ', pos9);
```

Jogo da Velha — Ler e Marcar

```
write('Jogador ', jogador, ' escolha (1-9): ');
readln(escolha);
if escolha = 1 then if pos1=' ' then pos1:=jogador else writeln('Ocupada!');
if escolha = 2 then if pos2=' ' then pos2:=jogador else writeln('Ocupada!');
{ ... até 9 }
```

Jogo da Velha — Verificar Vitória

```
vitoria := false;  
if (pos1=jogador) and (pos2=jogador) and (pos3=jogador) then vitoria := true;  
if (pos4=jogador) and (pos5=jogador) and (pos6=jogador) then vitoria := true;  
if (pos7=jogador) and (pos8=jogador) and (pos9=jogador) then vitoria := true;  
if (pos1=jogador) and (pos4=jogador) and (pos7=jogador) then vitoria := true;  
if (pos2=jogador) and (pos5=jogador) and (pos8=jogador) then vitoria := true;  
if (pos3=jogador) and (pos6=jogador) and (pos9=jogador) then vitoria := true;  
if (pos1=jogador) and (pos5=jogador) and (pos9=jogador) then vitoria := true;  
if (pos3=jogador) and (pos5=jogador) and (pos7=jogador) then vitoria := true;
```

Jogo da Velha — Empate e Alternância

```
if (jogadas = 9) and (not vitoria) then fim := true;  
if vitoria then fim := true;  
if not fim then  
    if jogador='X' then jogador:='0' else jogador:='X';
```

Jogo da Velha — Laço Principal Resumido

```
while not fim do
begin
{ mostrar }
{ ler + marcar }
jogadas := jogadas + 1;
{ verificar vitória / empate }
{ alternar }
end;
```

Próximos Passos (Além Desta Aula)

- Procedimentos e funções (modularização)
- Vetores e laços para reduzir repetição
- Validação robusta de entrada
- Estruturas de dados avançadas

Exercícios Propostos

1. Adaptar média para 4 notas.
2. Criar conversor de km/h para m/s.
3. Mostrar tabuada de um número (for ou while).
4. Extender jogo para pedir quem inicia.
5. Contar quantas vitórias cada jogador teve em 3 partidas.

Encerramento

Você viu a jornada completa: algoritmo → sintaxe Pascal → controle de fluxo → aplicação integradora simples.

Base pronta para evoluções futuras.

Bom estudo!

Gerar PDF

```
marp introducao_pascal.md --pdf
```

Ou usar extensão Marp no VS Code.