

FUNDAMENTOS DE PROGRAMAÇÃO

NOÇÕES DE LÓGICA

Maurício Moreira Neto¹

¹**Universidade Federal do Ceará**
Departamento de Computação

20 de fevereiro de 2020

Sumário

- 1 Objetivos
- 2 O que é lógica?
- 3 Lógica para Programação
- 4 Algoritmo
 - Representações de Algoritmos
 - Descrição Narrativa
 - Fluxograma
- 5 Tipos de Processamento
 - Teste de Mesa



Objetivos

- Apresentar os conceitos de elementos de lógica de programação e sua aplicação no cotidiano
- Definir algoritmo
- Relacionar lógica a algoritmos: lógica de programação
- Demonstrar o uso de algoritmos em situações do dia-a-dia
- Saber utilizar as diversas representações de algoritmos

Noções de Lógica

- “Lógica” é originária do grego *logos* = **Linguagem Racional**
- Ciência das formas do pensar, do raciocínio
- A lógica é o ramo da filosofia que **cuida das regras do bem pensar, ou do pensar correto**, sendo, portanto, um instrumento do pensar
- Segundo o *Michaelis*:
 - “Análise das formas e leis do pensamento, não se preocupando com a sua produção mas, somente, sua forma. Ou seja, a maneira que forma um pensamento e como é organizado e apresentado, possibilitando a uma conclusão”

Noções de Lógica

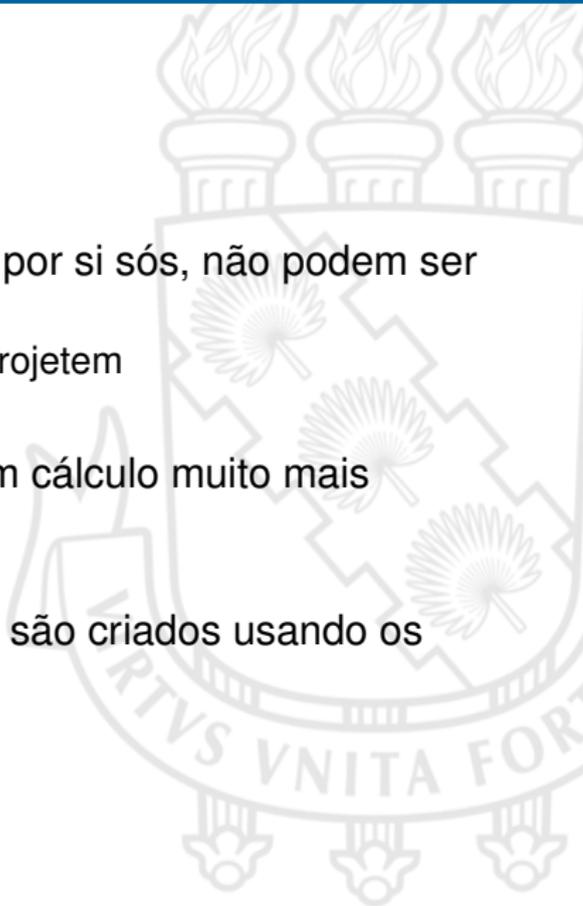
- **Silogismo:** a partir de duas **premissas**, podemos chegar a uma **conclusão**
 - 1 Todo mamífero é um animal
 - 2 Todo cavalo é um mamífero
 - 3 Portanto, todo cavalo é um animal
- Lógica no dia-a-dia
 - Sempre que quisermos colocar ordem no pensamento, estamos usando a lógica

Noções de Lógica

- O homem usa o raciocínio lógico para realizar suas atividades
- Sempre é estabelecido uma sequência adequada para realizar uma determinada tarefa
- **Exemplo:** Mudança de marcha de um veículo
 - 1 Coloca-se a marcha no ponto morto
 - 2 Passa-se a marcha na posição correta correspondente
 - 3 Após determinado valor de velocidade, muda-se a marcha novamente

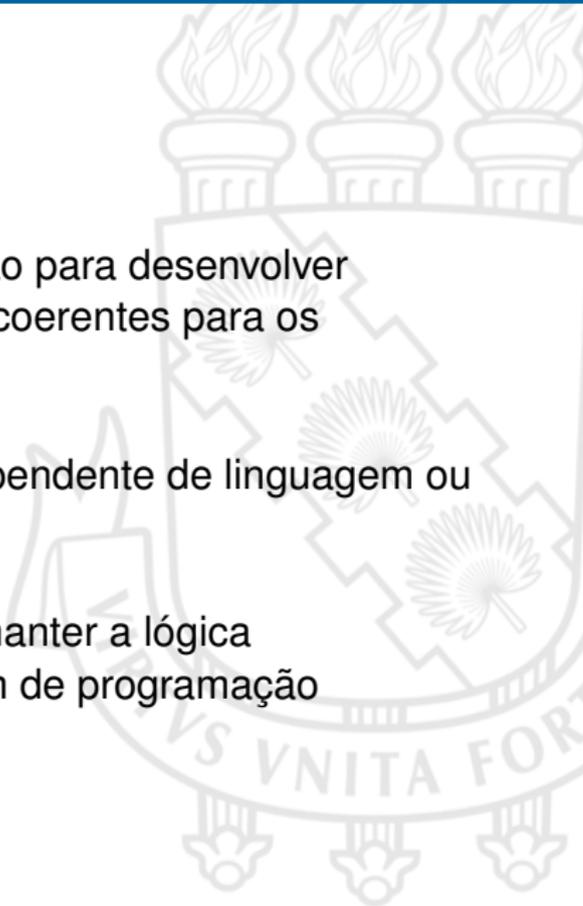
Lógica para Programação

- Computadores são máquinas e, por si sós, não podem ser inteligentes
 - É necessário que alguém as projete
- Um computador pode realizar um cálculo muito mais rápido que o cérebro humano
- Os programas de computadores são criados usando os conceitos de lógica



Lógica para Programação

- A lógica é usada na programação para desenvolver soluções logicamente válidas e coerentes para os problemas a serem resolvidos
- Raciocínio é algo abstrato, independente de linguagem ou idioma
- Algoritmos são utilizados para manter a lógica independente de uma linguagem de programação



Algoritmo

O que é um algoritmo?



Algoritmo

- É uma sequência finita de passos (ou instruções) realizados de maneira lógica que visam atingir um objetivo um objetivo bem definido
- A lógica é utilizada para ordenar os passos
- A sequência deve ser:
 - Finita
 - Não ambígua
- **Algoritmos estão presentes em nosso dia-a-dia**
 - Receitas culinárias
 - Instruções para realizar uma tarefa
 - Dirigir um determinado veículo

Exemplo de Algoritmos

- ... trocar uma lâmpada
- ... fazer um sanduíche
- ... fazer um omelete
- ... instalar um DVD
- ... tirar o carro da garagem
- ... sacar dinheiro do caixa eletrônico
- ...

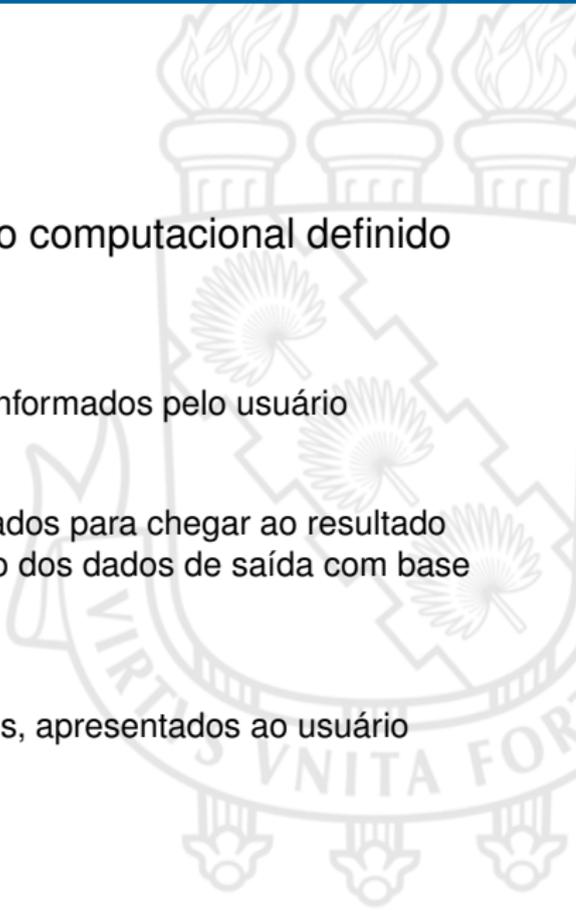


Exemplo de Algoritmos

- Sacar dinheiro em um caixa eletrônico
- 1 Ir até um caixa eletrônico
- 2 Inserir cartão do caixa eletrônico
- 3 Retirar cartão do caixa eletrônico
- 4 Escolher opção de saque
- 5 Digitar valor a ser sacado
- 6 Digitar código de letras
- 7 Se o saldo for maior ou igual à quantia desejada e a quantia desejada for menor ou igual ao limite diário disponível, sacar; senão, mostrar mensagem de indisponibilidade de saque

Algoritmos

- Um algoritmo é um procedimento computacional definido composto por 3 partes
 - **Entrada de dados**
 - São os dados do algoritmo informados pelo usuário
 - **Processamento de dados**
 - São os procedimentos utilizados para chegar ao resultado
 - É responsável pela obtenção dos dados de saída com base nos dados de entrada
 - **Saída de dados**
 - São os dados já processados, apresentados ao usuário



Método para Construção de Algoritmos

- 1 Compreender completamente o problema**
- 2 Definir os dados de entrada, as condições iniciais, que dados serão fornecidos e quais objetos fazem parte do problema
- 3 Definir os dados de saída**
- 4 Definir o processamento, cálculos a serem efetuados, restrições. Transformação dos dados de entrada em dados de saída. Identificar as responsabilidades
- 5 Construir o algoritmo utilizando técnicas descritas a seguir**
- 6 Testar o algoritmo usando simulações

Tipos de Representações de Algoritmos

- Descrição Narrativa
- Fluxograma
- Pseudo-código ou Portugol



Descrição Narrativa

- Consiste em analisar o enunciado do problema e escrever os passos a serem seguidos para a resolução utilizando **uma linguagem natural** (português, por exemplo)
- **Vantagem**
 - A linguagem é conhecida e de fácil compreensão
- **Desvantagem**
 - Ambiguidade, múltiplas interpretações, imprecisão nas instruções
- **Exemplo:** o programador disse ao amigo que o seu programa era o melhor...

Exemplo 1: Descrição Narrativa

- Algoritmos para mostrar o resultado da multiplicação de dois números

```
Passo 1: Receber os dois números que serão multiplicados
Passo 2: Multiplicar os números
Passo 3: Mostrar o resultado obtido na multiplicação
```

Exemplo 2: Descrição Narrativa

- Faça um algoritmo para mostrar o resultado da divisão de dois números

Passo 1: Receber os dois números que serão divididos

Passo 2: Se o segundo número for igual a zero, não será possível ser feita a divisão. Caso contrário, dividir os números e mostrar o resultado da divisão

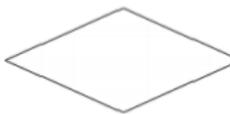
Fluxograma

- Consiste em analisar o enunciado do problema e escrever os passos a serem seguidos para a sua resolução utilizando **símbolos gráficos predefinidos**



- Entrada de Dados
- Linhas de fluxo: indica sequência das etapas e a direção do fluxo
- Saída de Dados

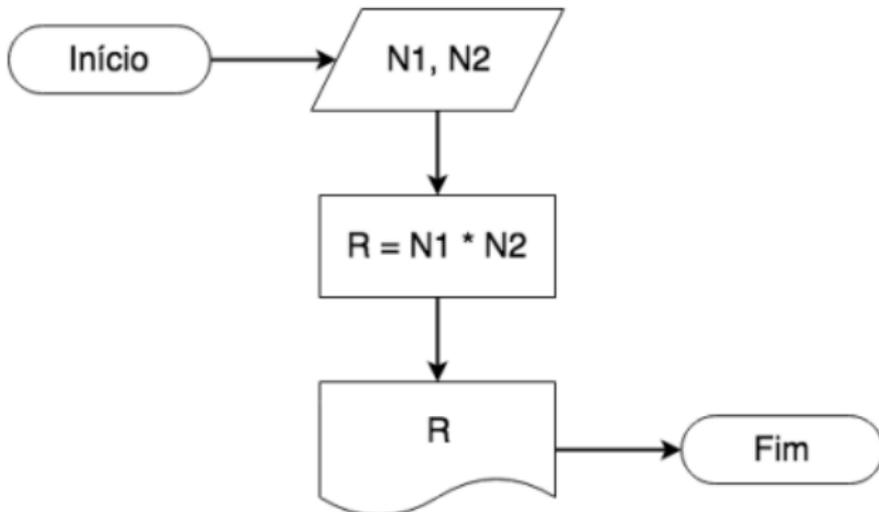
Fluxograma



- Início/Fim: Marca o início ou fim de um programa
- Decisão: indica desvios na sequência lógica de execução do programa
- Processamento: qualquer operação com alteração no conteúdo de uma variável

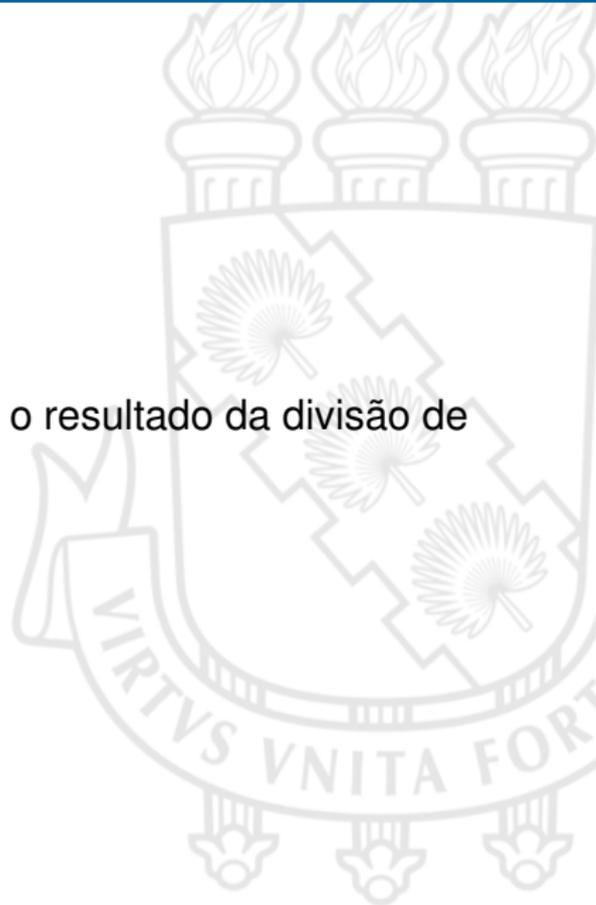
Exemplo 1: Fluxograma

- Faça um algoritmo para mostrar o resultado da multiplicação de dois números



Exemplo 2: Fluxograma

- Faça um algoritmo para mostrar o resultado da divisão de dois números



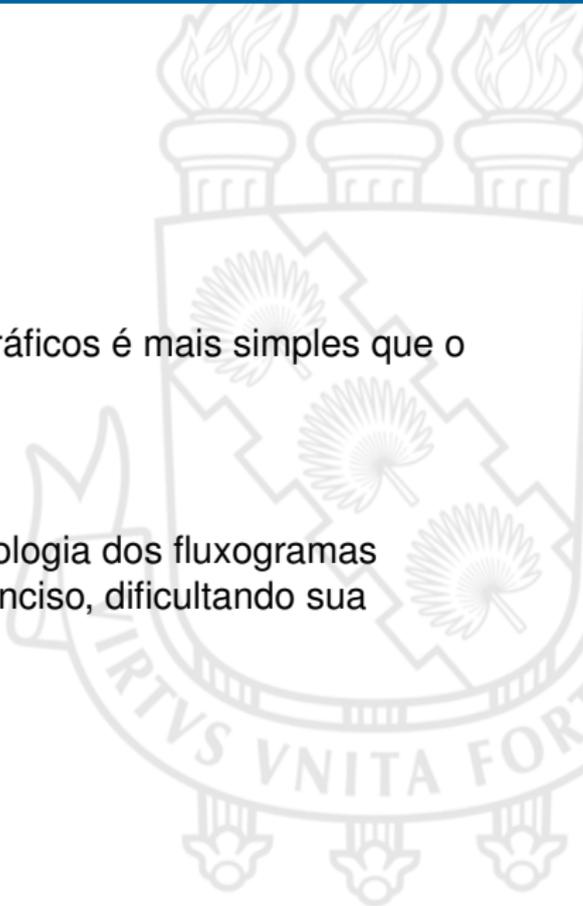
Fluxograma

■ Vantagem

- Entendimento de elementos gráficos é mais simples que o entendimento de textos

■ Desvantagem

- É necessário aprender a simbologia dos fluxogramas
- Fluxograma pode ser muito conciso, dificultando sua transição para um programa



Pseudo-código

- Consiste em analisar o enunciado do problema e escrever os passos a serem seguidos para a sua resolução por meio de **regras bem definidas**
- **Vantagem**
 - A transição do algoritmo para qualquer linguagem é quase imediata, bastando conhecer as palavras reservadas dessa linguagem que serão utilizadas
- **Desvantagem**
 - É preciso aprender as regras do pseudo-código

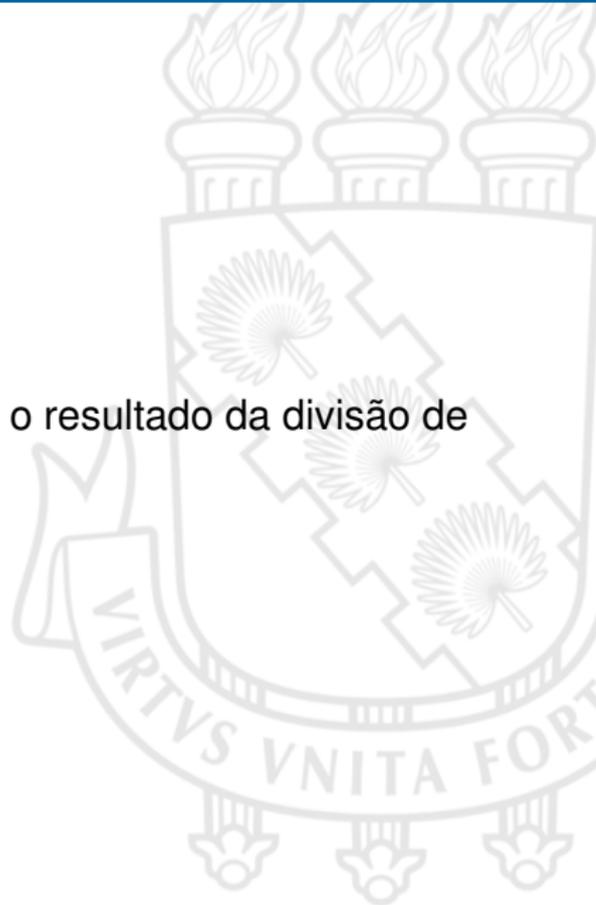
Exemplos 1: Pseudo-código

- Faça um algoritmo para mostrar o resultado da multiplicação de dois números

```
algoritmo
declare N1, N2, M: numérico
escreva("Digite dois números")
leia(N1)
leia(N2)
M = N1 * N2
escreva("Multiplicação = ", M)
fim algoritmo
```

Exemplos 2: Pseudo-código

- Faça um algoritmo para mostrar o resultado da divisão de dois números

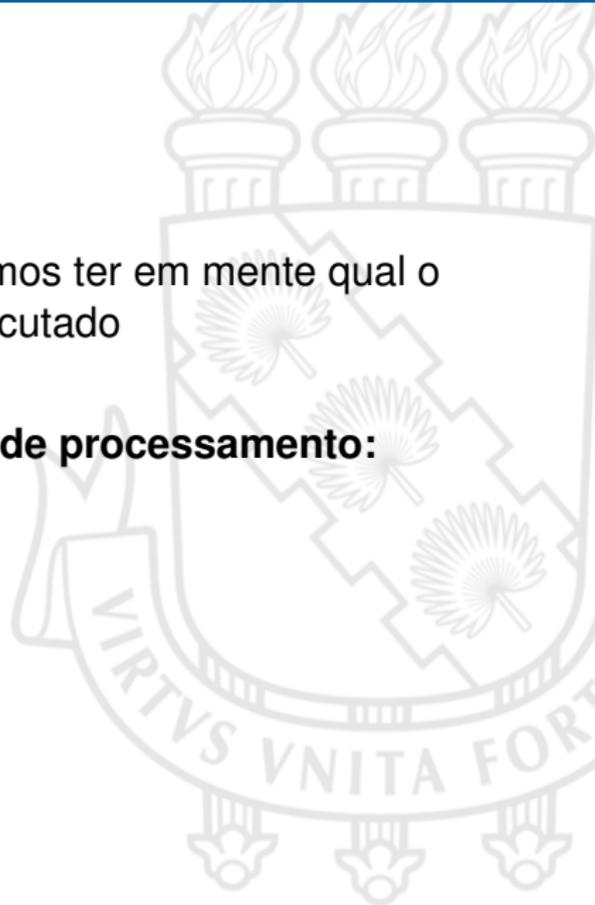


Exercícios

- Faça um algoritmo para calcular a média aritmética entre duas notas de um aluno e mostrar sua situação, que pode ser aprovado ou reprovado, utilizando:
 - Descrição narrativa, fluxograma e pseudo-código
- Faça um algoritmo para dizer se um número é par ou ímpar. Dica: % indica o resto da divisão:
 - Descrição narrativa, fluxograma e pseudo-código
- Faça um algoritmo para converter uma temperatura dada em Celsius para Fahrenheit, utilizando:
 - Descrição narrativa, fluxograma e pseudo-código

Tipos de Processamento

- Ao elaborar um algoritmo, devemos ter em mente qual o tipo de processamento será executado
- **Basicamente, existem 3 tipos de processamento:**
 - Processamento Sequencial
 - Processamento Condicional
 - Processamento de Repetição
 - Repetição determinada
 - Repetição indeterminada



Tipos de Processamento

■ Processamento Sequencial

- Instruções são executadas uma após a outra
- Não existe desvio na sequencia das instruções
- Cada instrução é executada uma única vez

■ Exemplo:

- Imprimir a media aritmética de duas notas

```
algoritmo media
declare nota1, nota2, media : numérico
leia(nota1)
leia(nota2)
media = (nota1 + nota2)/2
escreva(media)
fim algoritmo
```

Tipos de Processamento

- Processamento Sequencial
 - A ordem das instruções é importante!

```
algoritmo media
declare nota1, nota2, media : numérico
Leia (nota1)
Leia (nota2)
Imprima (media)
media = (nota1 + nota2)/2
```

```
algoritmo media
declare nota1, nota2, media : numérico
Leia (nota1)
Leia (nota2)
media = (nota1 + nota2)/2
Imprima (media)
```



Tipos de Processamento

■ Processamento condicional

- Um conjunto de instruções (pode ser apenas uma) que pode ou não ser executado
- Depende de uma condição
- Se a condição testada for verdadeira, o conjunto de instruções é executado



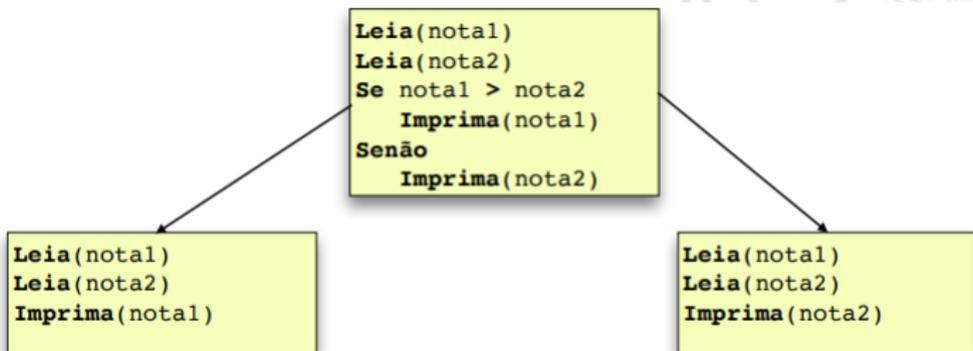
Tipos de Processamento

■ Processamento Condicional

- As instruções executadas dependem da situação

■ Exemplo:

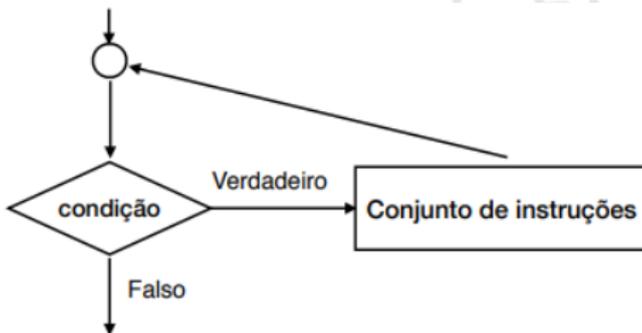
- Imprimir a maior dentre duas notas lidas



Tipos de Processamento

■ Processamento de Repetição

- Um conjunto de instruções (pode ser apenas uma) é executado um número definido ou indefinido de vezes
- Pode ser determinado por uma condição de parada
 - O conjunto de instruções é executado enquanto a condição for verdadeiro
 - O teste da condição é realizado antes de qualquer operação



Tipos de Processamento

■ Processamento com repetição

- Também chamado de laços condicionais
- Repetem um conjunto de comandos em seu interior

■ Exemplo:

- Imprimir a soma dos números inteiros de 1 a N
 - Soma = $1 + 2 + 3 + \dots + N$
 - Necessário identificar o que deve ser repetido no algoritmo

$$\text{Soma} = 1 \oplus 2 \oplus 3 \oplus \dots \oplus N$$

Teste de Mesa

- Após desenvolver um algoritmo, é preciso testá-lo!
- Uma maneira de fazer o teste é usando o **teste de mesa**
 - Basicamente, esse teste consiste em seguir as instruções do algoritmo de maneira precisa para verificar se o procedimento utilizado está correto ou não
 - Tentar utilizar um caso onde se conhece o resultado esperado
 - Permite reconstituir o passo a passo do algoritmo



Teste de Mesa

- Criar uma tabela de modo que:
 - Cada linha representa uma variável
 - As linhas correspondem as alterações naquela variável (de cima para baixo)

valor	N	soma

Teste de Mesa

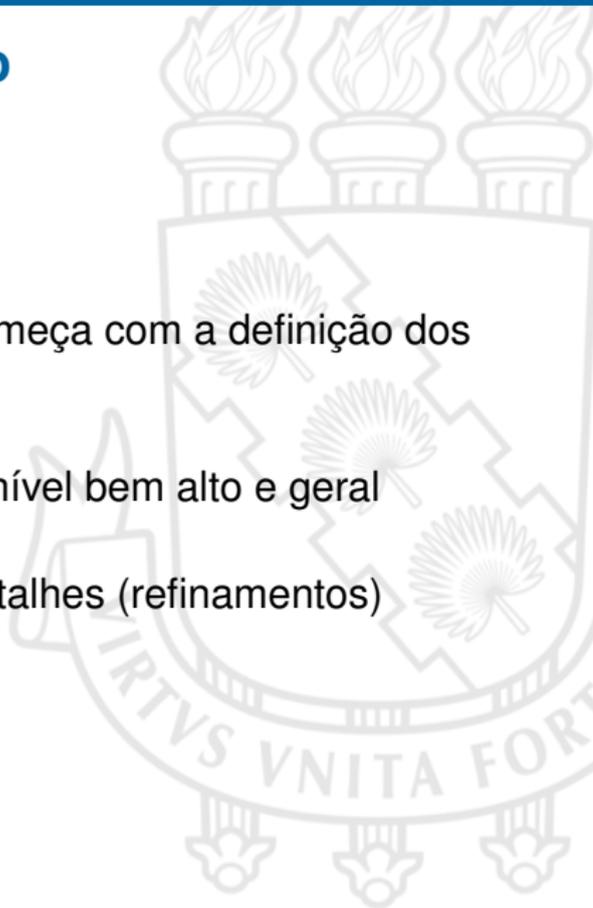
- **Exemplo:** Imprimir a média dos números positivos digitados. Deve-se parar quando um valor negativo ou zero for digitado
 - Valores digitados: 4, 2, 3 e -1
 - Média é 3

valor	N	soma
4	0	0
2	1	4
3	2	6
-1	3	9

```
N = 0
soma = 0
leia(valor)
enquanto valor > 0
soma = soma + valor
N = N + 1
leia(valor)
escreva(soma/N)
```

Metodologia de Programação

- A resolução de um problema começa com a definição dos dados e tarefas básicas
- Esta definição inicial é feita em nível bem alto e geral
- Não há preocupação com os detalhes (refinamentos)



Metodologia de Programação

■ Refinamentos Sucessivos (Top-Down)

- Consiste em pegar um grande problema, de difícil solução, e dividi-lo em problemas menores que devem ser mais facilmente resolvidos
 - Decompor uma ou várias tarefas em sub-tarefas mais detalhadas
 - É um processo iterativo, isto é, sub-tarefas podem ser decompostas em sub-tarefas ainda mais detalhadas

Algoritmo - Trocar um pneu furado

1. Levantar o carro parcialmente
2. Retirar o pneu furado
3. Instalar o novo pneu
4. Abaixar o carro

Algoritmo - Trocar um pneu furado

1. Retirar o estepe
2. Levantar o carro parcialmente
3. Retirar o pneu furado
4. Instalar o novo pneu
5. Apertar bem as porcas
6. Abaixar o carro

Refinamentos Sucessivos

Algoritmo - Trocar um pneu furado

1. Pegar as ferramentas no porta-malas
2. Retirar o estepe
3. Instalar o macaco
4. Levantar o carro parcialmente
5. Afrouxar os parafusos do pneu furado
6. Retirar o pneu furado
7. Instalar o novo pneu
8. Apertar bem as porcas
9. Abaixar o carro
10. Guardar o pneu furado e as ferramentas

Refinamentos Sucessivos

- O algoritmo proposto pode ainda ser refinado de várias outras formas
 - O que fazer se o macaco não estiver no porta-malas?
 - O que fazer se o estepe também estiver vazio?
 - Deve-se sempre puxar o freio de mão antes de executar estas operações
 - Limpar as mãos
 - Consertar o pneu furado
 - ...

Referências

- André Luiz Villar Forbellone, Henri Frederico Eberspächer, **Lógica de programação** (terceira edição), Pearson, 2005, ISBN 9788576050247.
- Ana Fernanda Gomes Ascencio e Edilene Aparecida Veneruchi de Campos, **Fundamentos de Programação de Computadores: Algoritmos, Pascal, C/C++ e Java** - 2 edição - Person.