Algoritmos – Estruturas Seqüenciais

José Gustavo de Souza Paiva

Introdução

- Objetivo básico da computação → auxiliar os seres humanos em trabalhos repetitivos e braçais, diminuindo esforços e economizando tempo
- O computador é capaz de auxiliar em qualquer coisa que lhe seja solicitada
  - Totalmente racional
  - Possui energia
- Entretanto
  - Não tem iniciativa
  - Nenhuma independência
  - Não é criativo nem inteligente
- Por isso, é necessário que ele receba suas instruções nos mínimos detalhes, para que tenha condições de realizar suas tarefas

Introdução

- Finalidade de um computador → receber, manipular e armazenar dados
- Todas essas operações são realizadas por meio de programas
- Estas tarefas constituem o PROCESSAMENTO DE DADOS
- Quando construímos um software para realizar determinado processamento de dados, devemos escrever um programa ou vários programas interligados
- Para que o computador consiga ler o programa e entender o que fazer, este programa deve ser escrito em uma linguagem que o computador entenda

Introdução

- Esta linguagem chama-se LINGUAGEM DE PROGRAMAÇÃO
- As etapas de desenvolvimento de um programa são
  - Análise → estuda-se o enunciado do problema para definir os dados de entrada, processamento e dados de saída
  - Algoritmo → utiliza-se ferramentas do tipo descrição narrativa, fluxograma ou português estruturado para descrever COMO resolver o problema identificado
  - Codificação → transforma-se o algoritmo em códigos na linguagem de programação escolhida

Algoritmo

- Definição
  - Seqüência de passos que visa atingir um objetivo bem definido
  - Esta seqüência de passos deve ser seguida para a realização de uma tarefa
- Algoritmos não são operações exclusivas de um computador
- A grande maioria das coisas que fazemos no dia-a-dia, fazemos por via de algoritmos
  - Somar três números
  - Fazer um sanduíche
  - Trocar uma lâmpada
  - Sacar dinheiro em um banco 24 horas
- IMPORTANTE → para a grande maioria dos problemas, é possível haver mais de um algoritmo de resolução

Representações de Algoritmos

- Descrição narrativa
  - Analisar o enunciado do problema e escrever, utilizando uma linguagem natural (língua portuguesa, por exemplo), os passos a serem seguidos para a resolução do problema
  - Vantagem → não é necessário aprender nenhum conceito novo
  - Desvantagem → línguas naturais são sempre passíveis de ambigüidades, ou seja, podem gerar múltiplas interpretações, e dificultar a posterior transcrição do problema em código

Representações de Algoritmos

- Fluxograma
  - Analisar o enunciado do problema e escrever, utilizando símbolos gráficos predefinidos, os passos a serem seguidos para a resolução dos problemas
  - Vantagem → entendimento de símbolos gráficos é mais fácil que entendimento de textos
  - Desvantagens
    - É necessário aprender a simbologia dos fluxogramas
    - Em alguns casos, o algoritmo resultante não apresenta muitos detalhes, dificultando sua transcrição para um programa

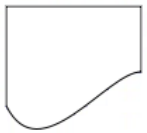
Fluxograma - Símbolos

Início e Fim do algoritmo

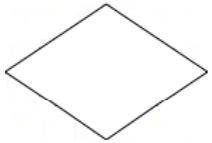
Sentido do fluxo de dados. Conecta símbolos ou blocos existentes

Processos em geral (cálculos ou atribuições de valores)

Entrada de dados



Saída de dados



Tomada de decisão, indicando a possibilidade de desvios

Representações de Algoritmos

- Pseudocódigo ou Portugol
  - Analisar o enunciado do problema e escrever, por meio de regras predefinidas, os passos a serem seguidos para a resolução do problema
  - Vantagem → a passagem para o código em linguagem de programação é quase imediata
  - Desvantagem → exige o aprendizado das regras do pseudocódigo

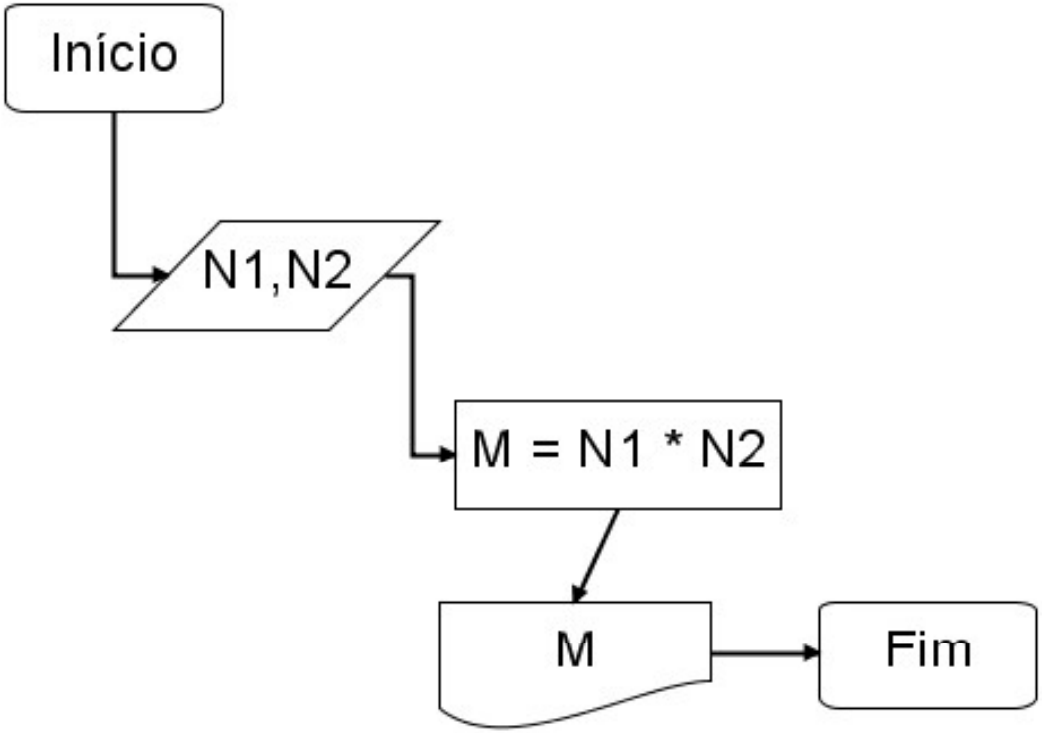
Exemplos de algoritmos

- Exemplo 1 → Faça um algoritmo para mostrar o resultado da multiplicação de dois números
- Algoritmo em descrição narrativa

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

Exemplos de Algoritmos

- Fluxograma



Exemplos de Algoritmos

- Pseudocódigo

```
ALGORITMO
    DECLARE N1,N2,M NUMÉRICO
    ESCREVA “Digite dois números:”
    LEIA N1, N2
    M ← N1 * N2
    ESCREVA “Multiplicação = ”, M
FIM
```

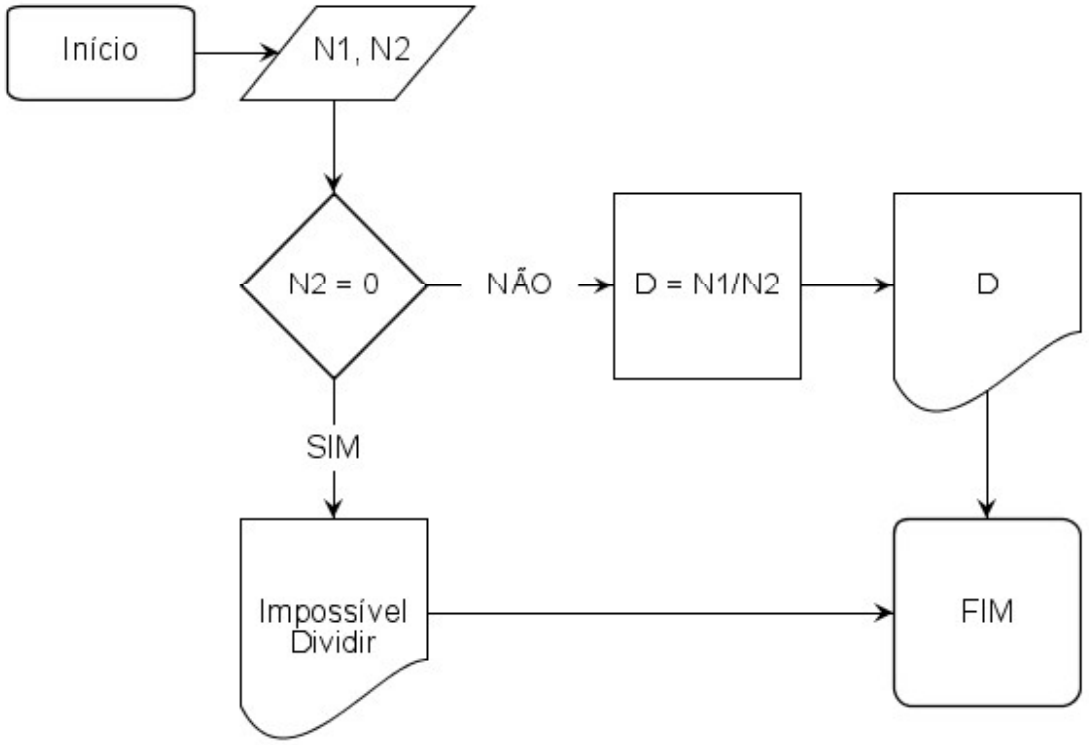
Exemplos de Algoritmos

- Exemplo 2 → Faça um algoritmo para mostrar o resultado da divisão de dois números
- Algoritmo em descrição narrativa

Passo 1: Receber os dois números que serão divididos
Passo 2: Se o segundo número for igual a zero, não poderá haver divisão, pois não existe divisão por zero; caso contrário, dividir os números e mostrar o resultado da divisão

Exemplos de Algoritmos

- Fluxograma



Exemplos de Algoritmos

- Pseudocódigo

```
ALGORITMO
DECLARE N1, N2, D NUMÉRICO
ESCREVA “Digite dois números:”
LEIA N1, N2
SE N2 = 0 ENTÃO
    ESCREVA “Impossível dividir.”
SENÃO
    INÍCIO
        D ← N1/N2
        ESCREVA “Divisão = ”,D
    FIM
FIM ALGORITMO
```

Exercício

- Faça um algoritmo para calcular a média aritmética entre duas notas de um aluno, e para mostrar a situação deste aluno, que pode ser APROVADO ou REPROVADO, considerando que a média de aprovação é 6
- Faça um algoritmo para calcular o novo salário de um funcionário. Sabe-se que os funcionários que possuem salário anual até R$500,00 terão aumento de 20%, os demais terão aumento de 10%

Variável

- Um algoritmo, e posteriormente um programa, recebe dados
- Tais dados precisam ser armazenados no computador, para que possam ser utilizados no processamento
- Este armazenamento é feito na memória
- Os dados são armazenados em locais específicos da memória, denominados ENDEREÇOS DE MEMÓRIA
- Assim, quando uma operação aritmética recebe dois operandos, cada operando é armazenado em um endereço de memória diferente, para ser utilizado no cálculo
- Como cada endereço de memória pode armazenar dados várias vezes, ou seja, seu conteúdo pode variar, chamamos estas posições de variáveis

Variável

- Cada variável representa, desta forma, uma posição de memória, e possui um nome e um tipo
- Seu conteúdo pode variar ao longo do tempo, durante a execução de um programa
- É importante ressaltar que, apesar de uma variável poder armazenar diversos valores durante a execução de um programa, ela só pode armazenar um valor a cada instante
- Todo computador possui uma tabela de alocação que contém o nome da variável, seu tipo (para determinar quantos bytes ocupará) e o seu endereço inicial de armazenamento
- Assim, quando queremos buscar algum dado na memória, basta saber o nome da variável, que o computador, por meio da tabela de alocação, vai buscá-lo automaticamente

Tipos de Dados

- Os tipos de dados determinam quantos bytes de memória uma determinada variável ocupará
- Os mais comuns são numérico, lógico e literal ou caractere
- Numérico
  - Podem representar números inteiros ou reais
  - Os números inteiros podem ser negativos ou positivos, e ocupam 2 bytes no computador
    - Desta forma, variam entre -32.767 e +32.768
  - Os números reais podem ser negativos ou positivos, e ocupam 4 bytes, e podem representar números com 6 a 11 dígitos significativos com sinal

Tipos de Dados

- Lógico
  - Também chamados de booleanos
  - Podem assumir os valores VERDADEIRO e FALSO
  - Ocupam apenas 1 byte de memória
- Literal ou caractere
  - Dados formados por um caractere, ou uma cadeia de caracteres
  - Ocupam um byte de memória por caractere

Identificadores

- Representam os nomes das variáveis, dos programas, constantes, rotinas, entre outros
- Regras básicas para formação de identificadores
  - Caracteres permitidos → números, letras maiúsculas ou minúsculas e o caractere sublinhado “_”
  - O primeiro caractere dever ser sempre uma letra ou o caractere sublinhado
  - Não são permitidos caracteres em branco e caracteres especiais (@,$,+,-,%,!)
  - Não é possível utilizar palavras reservadas nos identificadores, ou seja, palavras que pertencem à linguagem de programação utilizada

Identificadores

- Exemplos
  - A
  - a
  - Nota
  - nota
  - X5
  - nota_1
  - dia

Estrutura Seqüencial

- Estrutura básica para algoritmos em pseudocódigo

```
ALGORITMO
        DECLARE
        Bloco de comandos
FIM_ALGORITMO
```

Declaração de Variáveis

- Variáveis são declaradas após a palavra DECLARE, e os tipos mais utilizados são NUMÉRICO, LITERAL ou LÓGICO
- Exemplo

```
DECLARE
X NUMÉRICO
Y, Z LITERAL
teste LÓGICO
```

Atribuição

- Utilizado para atribuir um determinado valor ou operação à uma variável
- Representado pelo símbolo ←
- Exemplos
  - X ← 4
  - Y ← Y + 2
  - Z ← “aula”
  - Teste ← FALSO

Entrada/Saída de dados

- O comando de entrada é utilizado para receber dados informados pelo usuário
- Representado pela palavra LEIA
  - Exemplo → LEIA X (os dados informados pelo usuário serão armazenados na variável X)
- O comando de saída é utilizado para mostrar dados para o usuário, na tela do monitor, ou na impressora, entre outros
- Representado pela palavra ESCREVA
  - Exemplo → ESCREVA Y (mostra o valor armazenado na variável Y)
  - Exemplo → ESCREVA “Conteúdo de X é ”,X

Exercícios

1. Faça um programa que receba quatro números inteiros, calcule e mostre a soma desses números
2. Faça um programa que receba três notas, calcule e mostre a média aritmética entre elas
3. Faça um programa que receba três notas e seus respectivos pesos, calcule e mostre a média ponderada entre essas notas
4. Faça um programa que receba o salário de um funcionário, calcule e mostre o novo salário, sabendo-se que este sofreu um aumento de 25%
5. Faça um programa que receba o salário-base de um funcionário, calcule e mostre o salário a receber, sabendo-se que esse funcionário tem gratificação de 5% sobre o salário-base, e paga imposto de 7% sobre o salário-base
6. Faça um programa que calcule e mostre a área de um triângulo
7. Faça um programa que receba dois números inteiros e mostre qual é o maior deles

Exercícios

8. Faça um programa que receba o ano de nascimento de uma pessoa e o ano atual, calcule e mostre a idade desta pessoa, e quantos anos essa pessoa terá em 2020
9. Cada degrau de uma escada tem X de altura. Faça um programa que receba essa altura e a altura que o usuário deseja alcançar subindo a escada. Calcule e mostre quantos degraus o usuário deverá subir para atingir seu objetivo, sem se preocupar com a altura do usuário
10. Faça um programa que receba uma hora formada por hora e minutos, e calcule a hora digitada apenas em minutos

Referências

- ASCENCIO, A. F. G., CAMPOS, E. A. V., Fundamentos da Programação de Computadores, São Paulo, Prentice Hall, 2002