Introdução

Este capítulo introduz o funcionamento básico dos computadores e ensina como podemos utilizá-los e programá-los para realizar tarefas de nosso interesse.

Algoritmos

Antes estudamos que um algoritmo é um conjunto finito de instruções para realizar uma determinada tarefa. O conceito de algoritmos também é conhecido por outros nomes, entre eles podemos citar: receitas, procedimentos ou técnicas.

Recordando

As instruções que compõem o algoritmo são executadas seqüencialmente, ou seja, uma de cada vez, começando sempre pela primeira instrução. Em algum momento, durante a execução, deve-se encontrar uma instrução sinalizando o fim do algoritmo. Ou seja, o tempo necessário para terminar a execução do algoritmo deve ser finito e dentro de um prazo razoável.

Durante a execução, o algoritmo pode realizar saltos para outras instruções fora da seqüência. Por exemplo, voltar para a primeira instrução com objetivo de executá-las todas novamente. Se estes saltos estão associados com uma condição (ex: comparação entre dois números), então o algoritmo realiza as instruções dependendo se esta condição é verdadeira ou falsa.

> **Algoritmo** é uma seqüência finita de instruções necessárias para realizar uma determinada tarefa**.**

Para ser de utilidade, o algoritmo precisa produzir algum resultado. Caso contrário, o algoritmo não nos forneceria nenhuma informação nova e não existiria interesse pelo estudo do mesmo. Por exemplo, ele poderia calcular uma formula complexa ou localizar alguma informação em um banco de dados.

Algoritmos também precisam aceitar dados como entrada para guiar o resultado de sua execução. Senão, ele sempre produzirá a mesma resposta.

Exemplo

Um dos algoritmos mais conhecidos é o "Algoritmo de Euclides". Ele é utilizado para determinar o máximo divisor comum entre dois números.

O MDC é determinado realizando a divisão entre os dois números. Caso a divisão resulte em resto não nulo, a divisão é novamente realizada, substituindo os números anteriores pelo divisor e pelo resto.

Vamos utilizar o algoritmo para calcular o MDC entre 24 e 18, utilizando o algoritmo da transparência:

1. Leia dois números.
2. Divida o primeiro pelo segundo e guarde o resto.
3. Se o resto for 0 (zero),
   então escreva o segundo número e PARE.
4. Substitua o primeiro número pelo segundo.
5. Substitua o segundo número pelo resto da divisão.

```
Passo 1: Lê 21 e 15.
Passo 2: 21 ÷ 15 = 1, resto 6.
Passo 3: Resto é 0? Não, continua com passo 4.
Passo 4: Substitui 21 por 15.
Passo 5: Substitui 15 por 6.
Passo 6: Retornar para passo 2.
Passo 2: 15 ÷ 6 = 2, resto 3.
Passo 3: Resto é 0? Não, continua com passo 4.
Passo 4: Substitui 15 por 6.
Passo 5: Substitui 6 por 3.
Passo 6: Retornar para passo 2.
Passo 2: 6 ÷ 3 = 2, resto 0.
Passo 3: Resto é 0? Sim. Escreve 3 e PÁRA.
```

Ou seja, segundo o valor gerado pelo algoritmo, o MDC entre 21 e 15 é 3.
Note que:

- O algoritmo iniciou sua execução pela primeira instrução (passo 1) e realizou um número finito de instruções até chegar no PARE (total de mais 12 passos).
- A entrada do algoritmo foram os números 21 e 15. A saída foi 3.
- O passo 6 é uma instrução que faz um salto de volta para o passo 2, com o objetivo de continuar realizando novas divisões.
- O passo 3 verifica uma condição e executa a instrução somente se a condição for verdadeira.

Características de um algoritmo

A descrição das operações envolvidas em um algoritmo gera várias dificuldades quando utilizamos a língua portuguesa. Devemos considerar:

- A instruções não podem permitir **ambigüidades** na interpretação de sua operação. Alguém poderia questionar o que significa exatamente "Divida o primeiro número pelo segundo e guarde o resto". O que fazer com o quociente? Trata-se de uma divisão com casas decimais? Como proceder quando os números são negativos?
- Cada instrução deve realizar **uma única operação** bem definida. Na descrição do MDC, um dos passos diz: "escreva o segundo número e PARE". Claramente, a instrução PARE não está relacionada com a instrução ESCREVA.
- O algoritmo realiza apenas operações tecnicamente possíveis de executar dentro de um prazo de tempo razoável. Podemos utilizar a operação de divisão, pois existem circuitos elétricos capazes de determinar rapidamente o resultado da divisão de um número por um outro. No entanto, um algoritmo não poderia pedir a fatoração de um número em um único passo, pois não se conhece uma forma eficaz de realizar esta operação.

Linguagem de Programação

Para atender estas exigências, buscou-se outras formas mais objetivas para representar algoritmos.
A primeira alternativa seria uso de recursos gráficos, como, por exemplo, **diagramas**. No entanto, esta abordagem é muito trabalhosa e difícil de automatizar no computador. Assim, predominou uma forma mais prática, as **linguagens de programação**.

Elas possuem um vocabulário restrito (normalmente palavras do idioma inglês) e várias regras indicando seu único uso correto para escrever algoritmos. De forma geral, uma linguagem de programação impede que a pessoa escreva uma instrução de forma ambígua.

> A **linguagem de programação** permite a escrita formal de um **algoritmo.**

Existem programas de computador capazes de ler um algoritmo escrito em uma linguagem de programação e informar se as regras da linguagem foram violadas, impedindo assim qualquer possibilidade inconsistência ou ambigüidade.
Neste curso, estudaremos a linguagem de programação denominada "C".

Exemplo

Nesse ponto, limitar-nos-emos a estudar apenas algumas características do algoritmo escrito na linguagem de programação, pois estudaremos os detalhes durante o curso.

Notamos que as operações do algoritmo agora são expressas através de operações matemáticas indicadas por símbolos como ('=', '==', '%'). No entanto, os símbolos que representam as operações não necessariamente são os mesmo com os quais estamos acostumados.
Existem palavras com significado especial na linguagem de programação (chamadas palavras chave) tais como: 'int', 'scanf', 'do', 'break' e muitas outras que aprenderemos durante o curso.
O exemplo também ilustra um caso de regra de linguagem de programação C: todas as instruções devem ser separadas por ponto-e-vírgula.

```
int a, b, r;
scanf("%d %d", &a, &b);
do {
     r = a % b;
     if (r == 0) break;
     a = b;
     b = r;
};
printf("O MDC é %d", a);
return;
```

Algoritmo 2 – MDC na linguagem C

A linguagem de programação define um ponto inicial para a execução (neste caso, a primeira linha), comandos de entrada e saída de dados (respecitvamente `scanf` e `printf`) e comandos para terminar o algoritmo (**`return`**).

O computador

Os componentes que formam o computador podem ser classificados em duas categorias: **armazenamento** e **processamento**. O primeiro grupo envolve componentes que guardam dados a serem utilizados pelo computador. Alguns exemplos são: memórias, discos magnéticos, CDs e fitas magnéticas.
O segundo grupo é formados por componentes capazes de utilizar os dados armazenados no computador e realizar alguns tipos de operações sobre os mesmos, como localizar informações de um cliente, gerar relatórios, calcular estatísticas ou atualizar os dados armazenados.
Novos dados ou comandos para controlar o computador são obtidos através de dispositivos de "entrada" (teclado, *mouse*, conexões de rede, *scanner*, câmera digital, etc). Outros dispositivos, chamados dispositivos de "saída", permitem ao computador divulgar dados

obtidos pelo processamento ou contidos no armazenamento. Exemplos são: monitor, impressora, caixas de som.

> O computador **armazena** dados e realiza **operações** sobre os mesmos. Dados são obtidos por dispositivos de **entrada**. Resultados são informados em dispositivos de **saída**.

Instruções de Máquina

Para realizar operações sobre os dados contidos no armazenamento, o processamento do computador segue uma seqüência finita de instruções bem definidas e escritas em código de máquina. No dia a dia, nos referimos a estas instruções como aplicativos ou programas de computador.
Note como os programas de computador se assemelham aos conceitos que aprendemos durante o estudo de algoritmos.

> O computador é uma máquina **rápida** e **eficiente** para simular algoritmos.

Conseqüentemente, é natural escrever um algoritmo utilizando código de máquina, para que o computador realize a tarefa monótona de executar as operações de um algoritmo.

Exemplo

Apresentamos o algoritmo para encontrar o MDC de dois números, escrito especificamente para um computador de processador Intel Pentium.
As instruções são extremamente pouco intuitivas, pois estão longe de nossa capacidade de abstrair o problema. Ademais, cada operação exige várias instruções básicas para ser realizada, ou seja, uma descrição extremamente detalhista.
Para complicar ainda mais a descrição do algoritmo, o código de máquina exige instruções que só dizem respeito ao circuito eletrônico do processador. Neste caso, além da pessoa precisar preocupar-se em escrever corretamente o algoritmo, ela é forçada a planejar a execução do algoritmo respeitando as características de um determinado processador.

Níveis de Abstração

Durante esta introdução, apresentamos diversas opções para descrever um algoritmo. São elas: português, linguagem de programação, instrução de computador e código de máquina.
Entre elas, buscamos uma opção que permita escrever os algoritmos sem ambigüidade e com operações bem definidas.
Os dois extremos são português e código de máquina. A vantagem de se utilizar português está na facilidade de escrever o algoritmo com palavras de nosso dia a dia. No entanto, este texto será potencialmente ambíguo e não há ferramentas para traduzi-lo para a linguagem do computador.
Já a linguagem de máquina está justamente num formato que pode ser interpretado diretamente pelo computador. Além disso, as instruções são precisas e bem definidas, tal como desejado para descrever algoritmos. Infelizmente, é muito complicado escrever um algoritmo em código de máquina, pois o resultado é extremamente complexo e exige um

conhecimento profundo sobre as características do processador. Com isso, acaba-se perdendo tempo e energia no esforço com preocupações irrelevantes à programação de algoritmos.

Construção do Programa

A preparação de um programa de computador envolve vários passos. O objetivo final é escrever obter um arquivo com código de máquina que possa ser interpretado pelo computador como um programa ou aplicação.

1. Para isso, o programador deve primeiro elaborar um algoritmo considerando todas as questões estudadas anteriormente, principalmente: eficácia e corretude. O programador pode representá-lo conforme sua preferência.
2. Em seguida, o programador escreve o mesmo algoritmo utilizando uma linguagem de programação. Neste curso, aprenderemos a linguagem C para este propósito.
3. Depois, é necessário utilizar um compilador para traduzir o programa escrito em C para um conjunto de instruções que formam o código de máquina. Para isso, temos vários compiladores disponíveis. Este curso tem como foco principal o compilador C da Microsoft, chamado Visual C.net.
4. Depois de compilar o algoritmo, testamo-lo executando o programa no computador.

Eventualmente, ao escrever o algoritmo na linguagem de programação, cometemos erros que serão acusados pelo compilador. Neste caso, voltamos ao passo 2 para corrigir o algoritmos.

Durante a execução do programa, eventualmente será possível encontrar algumas inconsistências no seu comportamento. A origem do erro exigirá uma investigação minuciosa tanto do passo 1 como também do passo 2.