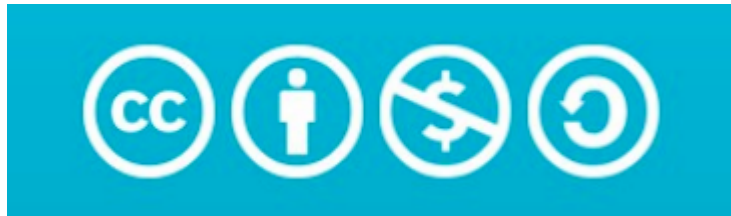


© 2017 by Bruno Luvizotto Carli



Esta obra é licenciada sob a Licença Creative Commons Atribuição–Não Comercial 2.5 Brasil. Você pode compartilhar e adaptar o conteúdo deste material desde que os devidos créditos sejam dados ao autor do trabalho. Para ver uma cópia desta licença, visite <http://creativecommons.org/licenses/by-ncsa/2.5/br/> ou envie uma carta para Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.

Índices para catálogo sistemático:

1. Computadores: Programação; 2. Programação de computadores; 3. Algoritmos; 4. Linguagem de Programação;

2° Edição

CAPA: Bruno L. Carli. Adaptado de: <https://pixabay.com/pt/animal-a-fotografiaanimal-close-up-1853944/>

Contato: brunolcarli@gmail.com

BRUNO LUVIZOTTO CARLI

2° Edição

Curitiba, PR

Edição do Autor

2018

Para meu pai, que me disse que computação era besteira...

Sumário

- Apresentação
- Capítulo 1
 - Conceito de Algoritmo
 - Algoritmos para a lógica de programação
 - Tipos de algoritmo
 - Python
- Capítulo 2
 - Construindo algoritmos com Python
 - Variáveis e tipos de dados
 - Variáveis em Python
 - Constantes
 - Expressões
 - Funções Intrínsecas
 - Entrada de dados
- Capítulo 3
 - Comentários
 - Estruturas Condicionais
 - Estrutura condicional simples
 - Estrutura condicional composta
 - Estrutura condicional aninhada
- Capítulo 4
 - Estruturas de repetição
 - Estrutura for
 - Estrutura while
 - Estruturas de repetição aninhadas
- Capítulo 5
 - Estruturas de dados
 - Estruturas de dados unidimensionais
 - Estruturas de dados multidimensionais
 - Tuplas
 - Dicionários
- Capítulo 6
 - Sub-rotinas e programação com arquivos
 - Sub-rotinas
 - Procedimentos
 - Funções

- Parâmetros
- Escopo das variáveis
- Recursividade
- Arquivos
- Bibliotecas e módulos
- Referências Bibliográficas

Apresentação

Olá amigo leitor, nesta obra pretendo elucidar conceitos básicos, porém fundamentais ao aprendizado da lógica de programação e o entendimento acerca da elaboração de algoritmos. Você será inicialmente apresentado à alguns exemplos de tipos de algoritmos e imerso na conceituação implícita do que é um algoritmo. Vamos ser apresentados à poderosa linguagem de programação Python, nossa principal ferramenta para implementação de algoritmos no decorrer deste livro.

Saliento desde já que esta obra não pretende ser mais um livro para o ensino da linguagem de programação Python, pretendo com este manuscrito apresentar os conceitos básicos e fundamentais dos algoritmos e lógica de programação como os tipos de dados, características de um algoritmo, estruturas de dados laços de repetição sub-rotinas e manipulação de arquivos permanentes, ao mesmo tempo em que possa induzir você leitor a conhecer e utilizar o Python para implementação dos nossos programas, e se você caro leitor se identificar positivamente com o Python, insisto que busque na farta bibliografia literária disponível abordando os mais diversos conceitos técnicos sobre Python.

Este livro objetiva orientar e colaborar no estudo dos algoritmos e lógica de programação de iniciantes no estudo da computação, hobbystas iniciantes e alunos de cursos técnicos e graduações que estejam começando nesta magnífica área que é a programação de computadores. Ressalto que este não é um livro orientado ao ensino de Python e espera-se de você leitor que compreenda os fundamentos da lógica de programação podendo facilmente adaptar os exemplos demonstrado nesta obra em outras linguagens de programação sem muita dificuldade.

- O autor.

Capítulo 1

Conceito de Algoritmo

Quando falamos em algoritmo é comum pensar em algo relacionado à matemática, o que de certa forma não está errado, pois quando pequenos na escola primária éramos incitados a resolver diversas contas utilizando algoritmos matemáticos necessários à resolução dos exercícios. Puga e Rissetti (2010, p. 9) comentam que “A matemática clássica é, em grande parte o estudo de determinados algoritmos”, isso pode a uma primeira vista assustar iniciantes no estudo da lógica de programação, mas vale salientar que não necessariamente o estudo de algoritmos envolve o estudo da matemática.

Para Puga e Rissetti (Op cit.) um algoritmo nada mais é que “uma sequência lógica de instruções que devem ser seguidas para a resolução de um problema ou execução de uma tarefa”. Ascencio e Campos (2010, p.2) vão ainda mais profundamente na conceituação de algoritmos citando diversas fontes definindo o que são algoritmos, dentre algumas pode-se citar: “sequência de passos que visa atingir um objetivo bem definido” (FORBELLONE, 1999 apud ASCENCIO; CAMPOS, 2010), “sequência finita de instruções ou operações cuja execução, em tempo finito, resolve um problema computacional, qualquer que seja sua instância” (SALVETTI, 1999 apud ASCENCIO & CAMPOS, 2010).

Etimologicamente, de acordo com Abbagnano (2007, p. 27), a palavra Algoritmo deriva do nome de Mohammed alKhuwarizmi que foi um astrólogo e matemático árabe do século IX, responsável por introduzir o sistema de numeração indiano no Ocidente, esta notação durante a Idade Média tornou-se conhecida como algorísmos, sendo conhecida hoje pelo sistema numérico decimal que utilizamos no nosso dia a dia.

Para finalizar a conceituação de algoritmo de uma forma, simples e entendível, um algoritmo é um passo a passo ou ainda, uma sequência de instruções para se chegar a um determinado objetivo, simples assim. Para exemplificar vamos demonstrar um algoritmo de uma tarefa corriqueira que todos fazemos diariamente: Beber água.

Passo a passo para beber água:

1. Pegar um copo ou recipiente;
2. Colocá-lo sob a torneira ou filtro de água;
3. Abrir a torneira/filtro;
4. Quando o copo estiver suficientemente cheio, fechar a torneira/filtro;
5. Beber a água do copo/recipiente;

Simples, não é? Pode-se perceber que não temos nada de muito matemático nisso, é um simples passo a passo, ou ainda, uma sequência finita de instruções que são seguidas para realizar uma tarefa (beber água). Vale-se ressaltar que não existe uma única forma de beber água, poderia-se beber água da fonte ou de um rio utilizando as mãos ou outros procedimentos, e isso é válido para todos os algoritmos, pois podem haver diferentes formas de realizar uma mesma tarefa desde que respeitem a lógica do processo, a exemplo disso pode-se dizer que não seria coerente fechar a torneira para depois colocar o copo abaixo da torneira para encher. Vamos demonstrar um outro exemplo de algoritmo: Uma ligação telefônica.

Passo a passo para realizar uma ligação:

1. Pegar o telefone;
2. Inserir o número para quem deseja-se telefonar;
3. Aguardar ser atendido;
4. Depois de realizar a conversaç o, colocar o telefone no gancho (ou apertar o bot o de desligar no caso de alguns telefones);

Algoritmos para a l gica de programa o

Certo, j  sabemos o que s o algoritmos e de onde essa palavra surgiu. Mas e a rela o entre os algoritmos e a programa o de computadores?

Os computadores n o compreendem instru es humanas como estas citadas anteriormente, os computadores s o conhecem um tipo de linguagem: bin ria (zeros e uns) ou como define Tanenbaum (2010, p. 1) linguagem de m quina, mas se acalme, n o h  necessidade de se tornar um expert em linguagem bin ria para programar um computador, para isso existem as linguagens de alto n vel ou linguagens de programa o, que de acordo com Pressman (2010, p. 677) "s o ve culos de comunica o entre os seres humanos e os computadores", as quais permitem que o programador possa elaborar um roteiro de instru es passo a passo (algoritmo) para que o computador execute uma determinada tarefa, estas linguagens de alto n vel ou popularmente conhecidas como linguagens de programa o, das quais pode-se citar o C, C++, Java, Cobol, Fortran, Python, dentre muitas outras linguagens existentes. Cada linguagem possui sua pr pria estrutura sint tica por m sua ess ncia reside na coer ncia l gica em que as instru es s o fornecidas ao computador, sendo assim, o algoritmo escrito em uma linguagem de programa o recebe o nome de programa (TANENBAUM, 2010).

Tipos de algoritmo

Os algoritmos podem ser descritos em diferentes formas, Ascencio e Campos (2010, p.3-4) apresentam a Descri o Narrativa, o Fluxograma e o Pseudoc digo. Cada um desses ser  analisado nesta se o:

Descrição Narrativa

A descrição narrativa já foi apresentada no início deste capítulo, nos exemplos de beber água e no exemplo de realizar uma ligação telefônica. Consiste em descrever em linguagem natural a sequência de eventos que se sucederão. Vamos exemplificar com mais um algoritmo em descrição narrativa:

Somar dois números:

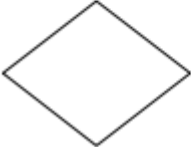
1. Definir o primeiro número;
2. Definir o segundo número;
3. Realizar a operação de soma entre os dois números;
4. Obter e demonstrar o resultado;

Esse tipo de algoritmo nos lembra muito de receitas de bolo por exemplo, porém possui uma grande desvantagem, por se tratar de uma linguagem natural, suas instruções podem ser ambíguas abrindo espaço para diferentes interpretações, dificultando sua transcrição em programa(ASCENCIO; CAMPOS, 2010), fazendo com que este modelo de algoritmo não seja muito utilizado em programação.

Fluxograma

O fluxograma é uma forma de representar a sequência de instruções através de figuras geométricas, sendo elas:

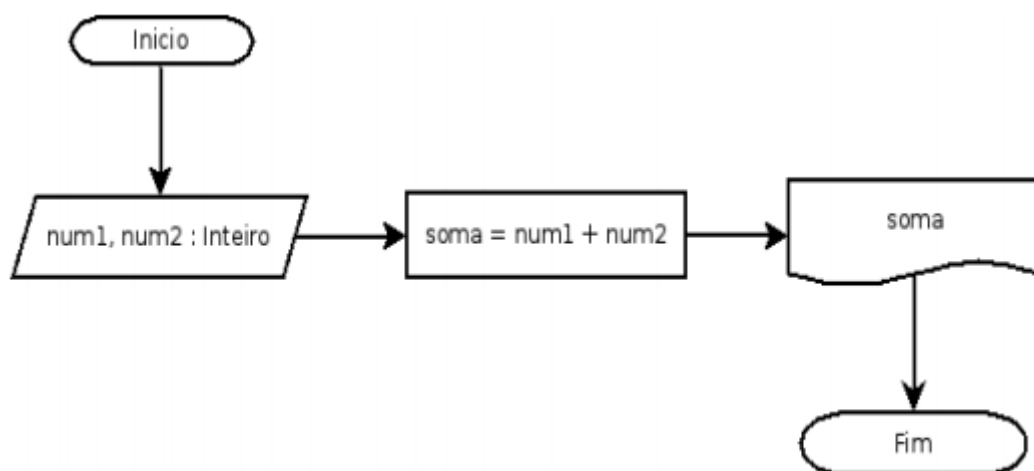
Tabela 1 – Símbolos utilizados no fluxograma e seus significados

	Símbolo utilizado para indicar o início e o fim do algoritmo.
	Permite indicar o sentido do fluxo de dados. Serve exclusivamente para conectar os símbolos ou blocos existentes.
	Símbolo utilizado para realizar cálculos e atribuições de valores.
	Símbolo utilizado para representar a entrada de dados.
	Símbolo utilizado para representar a saída de dados.
	Símbolo utilizado para indicar que deve ser tomada uma decisão, apontando a possibilidade de desvio.

Fonte: Ascencio & Campos (2010, p. 4)

Vamos exemplificar a utilização do fluxograma em um algoritmo cujo objetivo é realizar a soma de dois números, o mesmo algoritmo apresentado anteriormente:

Figura 1. Fluxograma representando um algoritmo para soma de dois números



Fonte: O autor.

As desvantagens do fluxograma segundo Ascencio e Campos(2010) são a necessidade de aprender os significados dos símbolos do fluxograma, que muitas vezes podem diferir, como por exemplo podemos ver na Figura 2 duas representações diferentes para o símbolo de saída de dados. Outra desvantagem apresentada é que o fluxograma não apresenta muitos detalhes, o que também dificulta a implementação deste algoritmo em um programa de computador.

Figura 2: Representações para a Saída de Dados



Fonte: Figura do lado esquerdo, Ascencio & Campos (2010). Figura do lado direito, Puga & Rissetti (2010).

Pseudocódigo

O Pseudocódigo, também conhecido como Portugol (ASCENCIO; CAMPOS, 2010) ou Português Estruturado (PUGA; RISSETTI, 2010) é um modelo de algoritmo muito semelhante a um código escrito em linguagem de programação de alto nível, tornando sua implementação em um programa uma tarefa muito simples para o programador. Para Puga e Rissetti (2010, p. 11) este é uma representação de algoritmo em uma linguagem intermediária entre linguagem de programação e a linguagem natural humana, mais especificamente, o nosso Português. De acordo com Puga & Rissetti (Op. Cit.) o nome "Pseudocódigo" significa "falso código", justamente por ser extremamente parecido com um algoritmo implementado em linguagem de alto nível. Vamos ver o mesmo exemplo anterior (soma de dois números) representado em Pseudocódigo:

```
ALGORITMO: soma
VAR
    num1, num2, soma : Inteiro
INICIO
    LEIA(num1)
    LEIA(num2)
    soma ← num1 + num2
    ESCREVA(soma)
FIM.
```

Este exemplo pode ser facilmente transcrito em qualquer linguagem de programação e executado por um computador. Vamos analisar a estrutura do pseudocódigo:

Tabela 2 – Estrutura do pseudocódigo

Psudocódigo	Descrição
ALGORITMO: soma	Esta primeira seção declara o nome do algoritmo
VAR: num1, num2, soma : inteiro	Esta seção é chamada de declaração de variáveis, onde todas as variáveis que serão utilizadas no algoritmo devem ser fornecidas, neste caso precisamos de três delas, o primeiro número, o segundo número e o resultado da soma. Todas elas são do tipo inteiro então declaramos uma após a outra, separando-as por vírgulas. Veremos mais sobre variáveis mais adiante.
INICIO LEIA(num1) LEIA(num2) soma ← num1 + num2 ESCREVA(soma) FIM	Esta seção é o corpo principal do algoritmo, onde todas as instruções serão declaradas. Neste caso temos duas instruções, LEIA() e ESCREVA(). A instrução LEIA() é um comando de entrada de dados, o que significa que um valor será fornecido para a variável num1 e num2. Então temos a atribuição do resultado para a variável soma, esta atribuição é representada pelo símbolo "←". O comando ESCREVA() representa uma saída de dados, que neste caso demonstra a exibição do valor guardado na variável "soma". FIM. Indica que chegamos ao fim do algoritmo.

Fonte: O autor.

Algumas desvantagens do pseudocódigo apresentadas por Ascencio e Campos (2010) é que deve-se aprender as regras do pseudocódigo, da mesma forma que se aprenderia os significados dos símbolos do fluxograma ou as regras sintáticas de uma linguagem de programação, além do mais não é possível implementar programas reais em pseudocódigo, por estes motivo vamos conceber a linguagem de programação Python como forma de implementação e aprendizado dos nossos algoritmos por sua facilidade de compreensão, ao final deste livro você não somente saberá os princípios básicos da construção e interpretação de algoritmos assim como terá os conhecimentos básicos de uma poderosa linguagem de programação e estará escrevendo seus próprios programas de computador.

Python

Python é uma linguagem extremamente eficiente: seus programas farão mais com menos linhas de código, se comparado ao que muitas outras linguagens exigiriam. A sintaxe de Python também ajudará você a escrever um código "limpo". Seu código será fácil de ler, fácil de depurar, fácil de estender e de expandir, quando comparados com outras linguagens. (MATTHES, 2016. p. 28)

Para a metodologia deste livro, pretende-se utilizar a Linguagem Python como forma de aprendizado de algoritmos, visto que as formas mais populares de algoritmos apresentam desvantagens relacionadas ao aprendizado das regras próprias de cada modelo, utilizaremos a