Noções de algoritmos de programação

O que são algoritmos?

Algumas definições

1. Um algoritmo é uma sequência de instruções ordenadas de forma lógica para a resolução de uma determinada tarefa ou problema.
2. O conceito central da programação e da Ciência da Computação é o conceito de algoritmos, isto é, programar é basicamente construir algoritmos.
3. Um algoritmo é uma receita para um processo computacional e consiste de uma série de operações primitivas, interconectadas devidamente, sobre um conjunto de objetos. Os objetos manipulados por essas receitas são as variáveis.
4. Serve como modelo para programas, pois sua linguagem é intermediária à linguagem humana e às linguagens de programação.

O que são algoritmos?

- Um **algoritmo** é formalmente uma sequência finita de passos que levam à execução de uma tarefa.

- Podemos pensar em **algoritmo como uma receita**, **uma sequência de instruções** que tem a função de atingir uma meta específica.

- Essa tarefa não pode ser redundante nem subjetiva na sua definição, devendo ser clara e precisa.

Algoritmo “Chupar uma bala”.

1. Pegar a bala;
2. retirar o papel;
3. colocar a bala na boca;
4. chupar a bala;
5. jogar o papel no lixo.

Algoritmo “Somar dois números quaisquer”.

1. Escreva o primeiro número;
2. escreva o segundo número;
3. some o primeiro número com o segundo e escreva o resultado.

Seu primeiro Algoritmo – mandar uma mensagem para um grupo de pessoas do WhasApp

Seu segundo Algoritmo – O cálculo do preço médio de suas ações na bolsa de valores

Seu segundo Algoritmo – O cálculo do preço médio de suas ações na bolsa de valores

Em janeiro, você comprou 1000 ações a um custo total de R$ 25. Em março, você comprou mais 2000 ações a R$ 27. Em abril, mais 3000 a R$ 26.

O preço médio é igual a (1000×25)+(2000×27)+(3000×26) dividido pelo número total de ações compradas que é 6000. No caso, o resultado é R$ 26,17.

Tipos ABC de sistemas

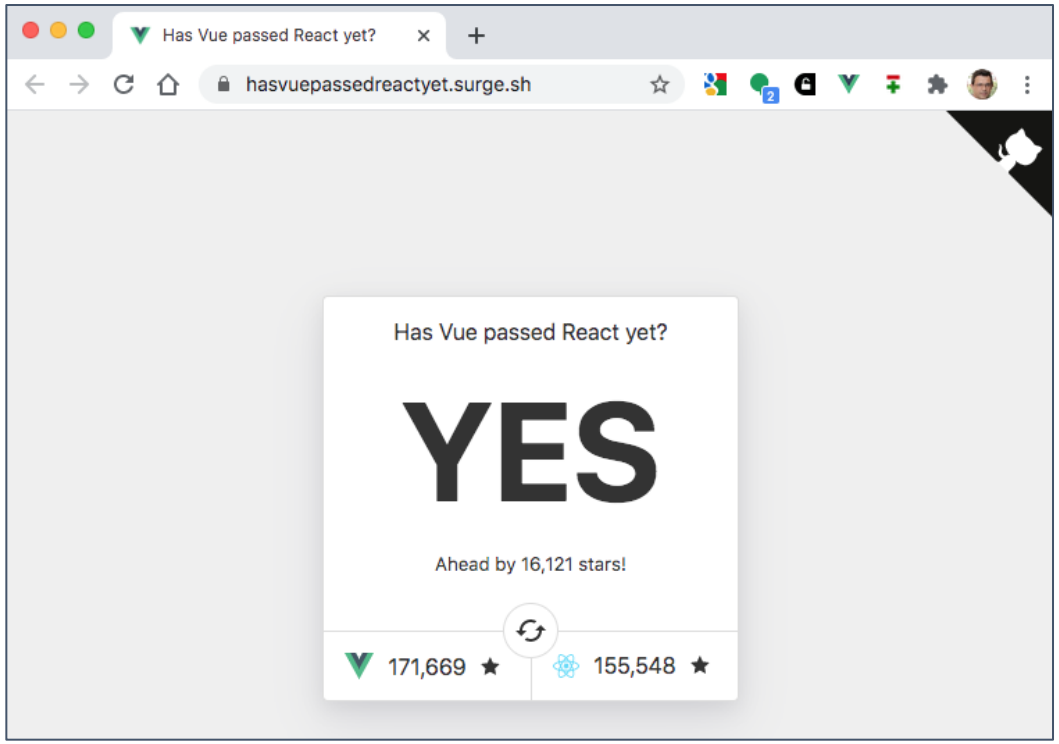
- Proposta por Bertrand Meyer
- https://bertrandmeyer.com/2013/03/25/the-abc-of-software-engineering/

- Três tipos de software:
  - Sistemas C (Casuais)
  - Sistemas B (Business)
  - Sistemas A (Acute)

Sistemas C (Casuais)

- Tipo muito comum de sistema
- Sistemas pequenos, sem muita importância
- Podem ter bugs; às vezes, são descartáveis
- Desenvolvidos por 1-2 engenheiros
- **Não se beneficiam dos princípios e práticas deste curso**
- Risco: "over-engineering"

Exemplo de Sistema Casual



https://hasvuepassedreactyet.surge.sh

Sistemas B (Business)

- Sistemas importantes para uma organização
- **Sistemas beneficiam-se do que veremos no curso**
- Risco: não usarem técnicas de ES e se tornarem um passivo, em vez de um ativo para as organizações

Sistemas A (Acute)

- Sistemas onde nada pode dar errado, pois o custo é imenso, em termos de vidas humanas e/ou $$$
- Também chamados sistemas de missão crítica

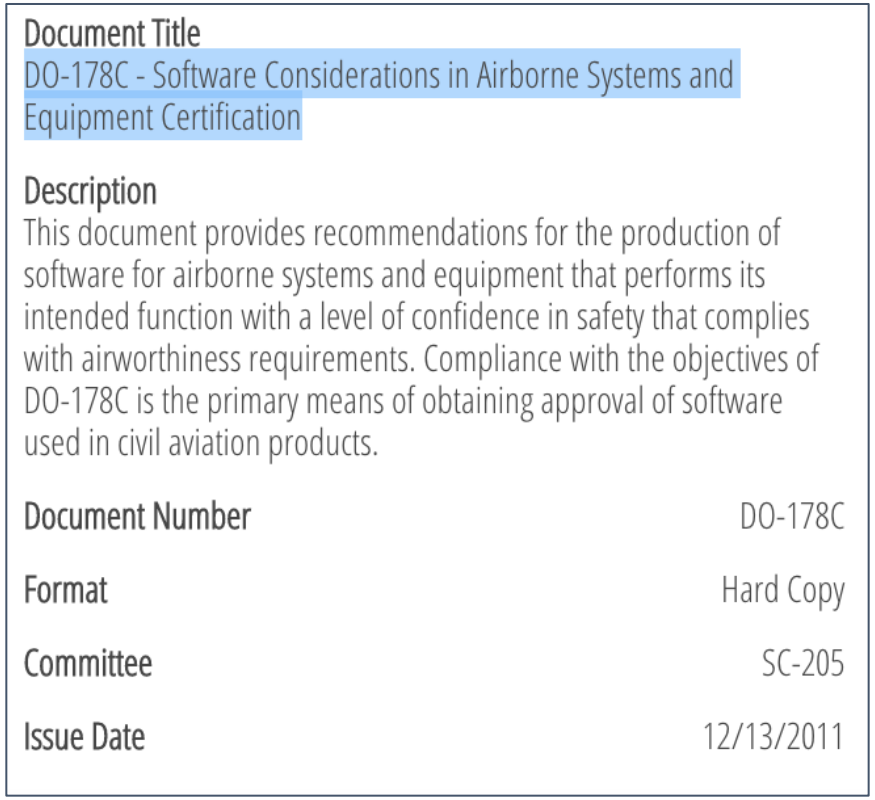
Metrô

Aviação

Medicina

Sistemas A (Acute)

- Requerem certificações
- **Fora do escopo do nosso curso**



Document Title
DO-178C - Software Considerations in Airborne Systems and Equipment Certification

Description
This document provides recommendations for the production of software for airborne systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements. Compliance with the objectives of DO-178C is the primary means of obtaining approval of software used in civil aviation products.

Document Number	DO-178C
Format	Hard Copy
Committee	SC-205
Issue Date	12/13/2011

Olhando e revisando os exemplos que acabamos de citar, podemos perceber que tudo não passa de uma **sequência lógica ordenada de forma a se alcançar um objetivo específico**, com prazos para começar (início) e terminar (fim).

Para lembrar!

1. A **ordem lógica** da execução das tarefas é importante.
2. Todo algoritmo tem **início e fim**.
3. Um algoritmo tem que ser **completo**.
4. Um algoritmo deve ter um alto índice de **detalhamento**.
5. Cada tarefa ou etapa é chamada de **instrução**.

Mas não é só sair programando? Tenho que fazer isso?

Se fôssemos fazer analogias com a construção de uma casa, o **algoritmo** representaria a **fundação da casa**, assim como a **programação** propriamente dita representaria a **construção das paredes da casa**.

Até podemos levantar uma casa sem fundação, mas imagine os problemas que virão no futuro.

No mesmo raciocínio, até podemos construir programas sem algoritmos, mas, pode ter certeza, quanto maior for o seu programa, maiores serão suas dores de cabeça, principalmente se tiver que fazer manutenções e atualizações no programa. Você deixará de ser eficiente e eficaz e perderá mais tempo corrigindo problemas do que criando inovações.

Mas não é só sair programando? Tenho que fazer isso?

A resolução de qualquer problema por processos informáticos (e os sistemas web são processos informáticos) estrutura-se segundo três fases essenciais:

④. Análise do problema.
④. Desenho do algoritmo.
④. Codificação.

Formas de representação de um algoritmo

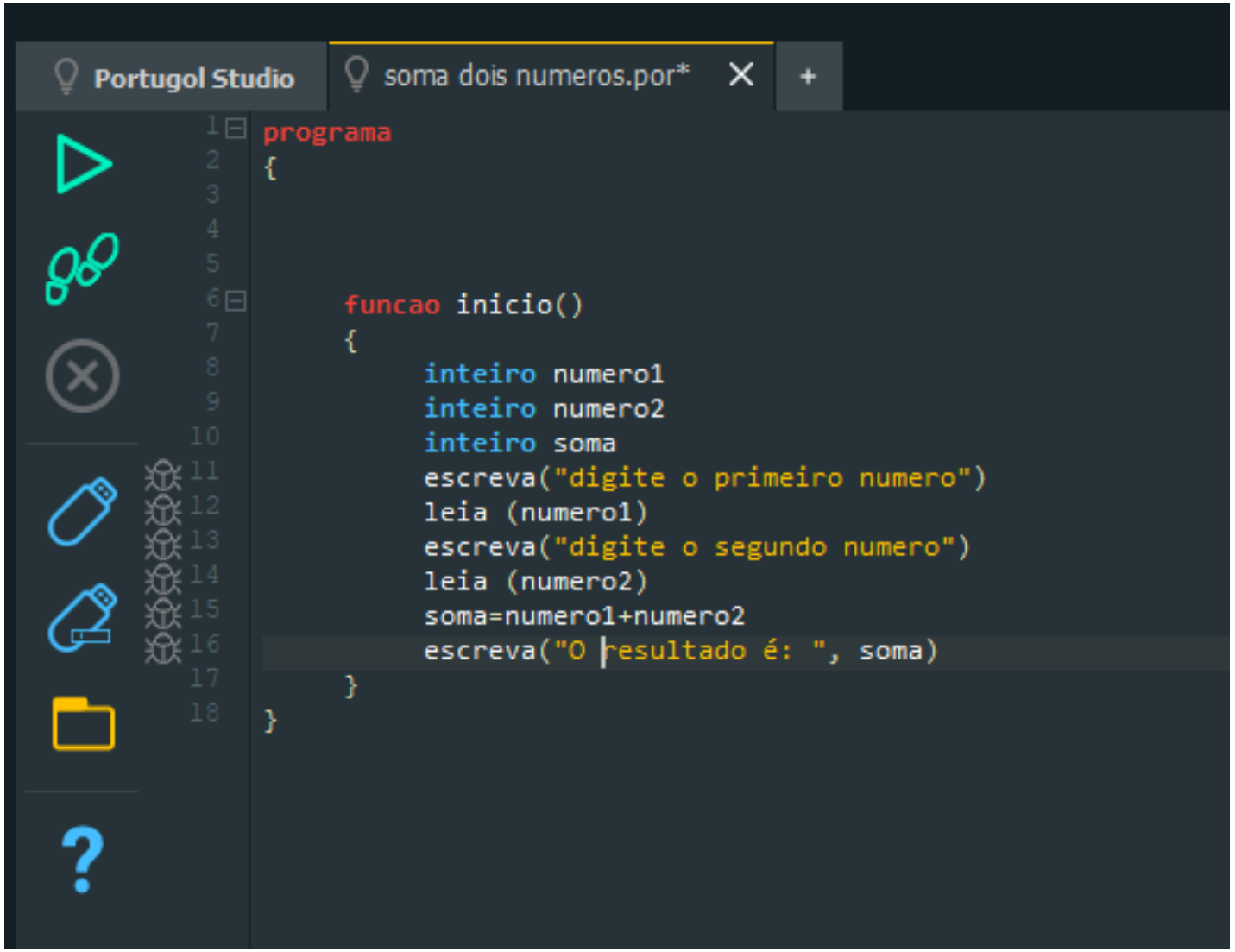
Vimos que os algoritmos podem ser representados de forma **descritiva**, mas também podem ser por **Pseudocódigo** e **fluxogramas**

Pseudocódigo

Pseudocódigo é um apelido dado a um código. A grande diferença entre pseudocódigo e código está relacionada à simplicidade, pois pseudocódigo utiliza palavras de nossa língua para representar ações, instruções, e é regido por poucas regras.

```
algoritmo <nomeDoAlgoritmos>
variáveis
    <nomeDaVariavel> : <tipoDaVariavel>
inicio
    <comando1>
    <comando2>
        ...
    <comandon>
fim
```

Pseudocódigo



A imagem ao lado apresenta um algoritmo
Em Portugol que pede dois número de entrada
Lê os números e realiza a soma desse números.
Ao final é apresentado o resultado

Pseudocódigo

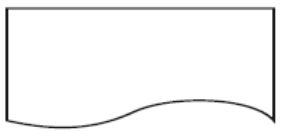
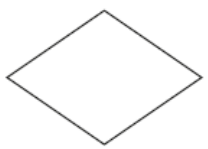
```
programa
{

    funcao inicio()
    {
        inteiro  numero1
        inteiro  numero2
        inteiro soma

        escreva("Digite o primeiro número: ")
        leia(numero1)

        escreva("Digite o segundo número: ")
        leia(numero2)
        soma=numero1+numero2
        escreva("O resultado é: ", soma)

    }
```

A imagem ao lado apresenta um algoritmo
Em Portugol que pede dois número de entrada
Lê os números e realiza a soma desse números.
Ao final é apresentado o resultado

Fluxograma

Utiliza figuras para representar as ações, instruções e seu fluxo.

Fluxograma

	Símbolo utilizado para indicar o início e o fim do algoritmo.
	Utilizado para conectar um símbolo a outro, indicando o sentido do fluxo de dados.
	Indica a realização de processos. Normalmente utilizado para realização de cálculos e atribuições.
	Símbolo utilizado para representar a entrada de dados.
	Símbolo utilizado para representar a saída de dados.
	Indica a realização de uma decisão, onde podemos tomar um determinado caminho caso o resultado da decisão seja verdadeiro ou, então, um outro caminho caso o resultado da decisão seja falso.

Fonte: Revisado e adaptado por Zanini (2011).

A figura ao lado apresenta cinco figuras geométricas que podem ser usadas na construção de fluxogramas

Fluxograma

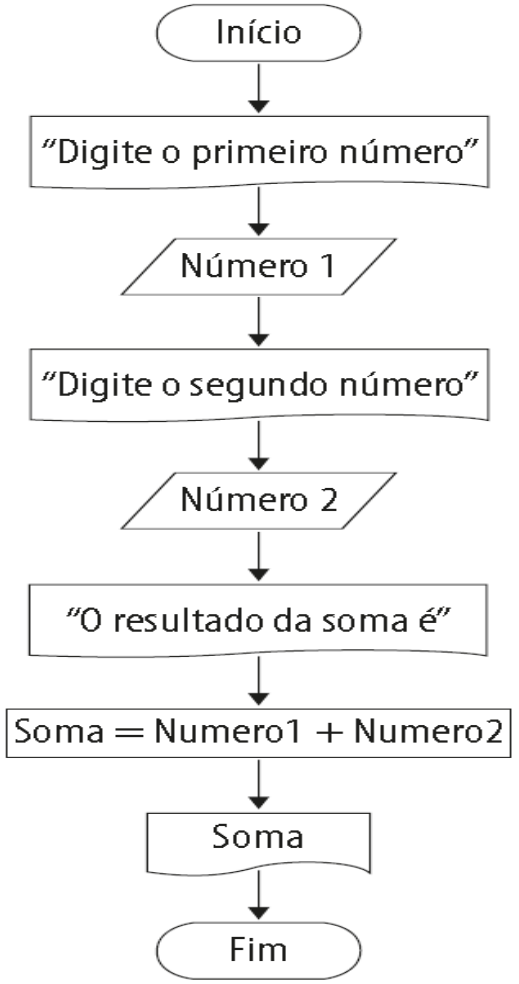


Figura 3.1 – Exemplo de fluxograma
Fonte: Revisado e adaptado por Zanini (2011).

A figura ao lado utiliza as figuras geométricas na construção do fluxograma da leitura de dois números, soma a apresentação dos resultados.

Fases para a construção do algoritmo

Ao montar um algoritmo, precisamos primeiro dividir o problema apresentado em três fases fundamentais:

Entrada, Processamento e Saída

ENTRADA DE DADOS ➔ PROCESSAMENTO ➔ SAÍDA DE DADOS

Fases para a construção do algoritmo

ENTRADA: São os dados de entrada do algoritmo, ou seja, os dados que o computador precisa ter para poder executar a ação, instrução, solicitada. Normalmente são as informações dadas ao computador através de algum dispositivo de entrada, (teclado, mouse), ou então geradas pelo próprio computador (resultado de uma outra operação, geração de número aleatório).

PROCESSAMENTO: São os procedimentos utilizados para chegar ao resultado final. Seria a execução da ação, instrução, propriamente dita.

SAÍDA: São os dados já processados, ou seja, o resultado obtido com a ação, instrução, executada. Normalmente esses são apresentados em um dispositivo de saída de dados (monitor, impressora) ou então guardados na memória do computador (variáveis).

Fases para a construção do algoritmo

Imagine o seguinte problema: calcular a média aritmética de um aluno.
O aluno realizou quatro provas: P1, P2, P3 e P4.
Onde:

Média Final = (P1 + P2 + P3 + P4) / 4

Para montar o algoritmo proposto, faremos três perguntas:

1. **Quais os dados de entrada?**
2. **Qual o processamento a ser utilizado?**
3. **Qual será o dado de saída?**

Fases para a construção do algoritmo

a) Quais são os dados de entrada?

R: Os dados de entrada são as notas das provas, ou seja, P1, P2, P3 e P4.

b) Qual será o processamento a ser utilizado?

R: Será somar todos os dados de entrada e dividi-los por 4 (quatro) (P1 + P2 + P3 + P4) / 4.

c) Qual será o dado de saída?

R: O dado de saída será a média final, ou seja, o resultado do cálculo feito pelo processamento.

Python: para que serve?

Roberto Fabiano Fernandes

O que é o Python?

O Python, é uma linguagem de programação.

Ok.... Mas o que é uma linguagem de programação??

Assim como temos diferentes línguas para falarmos, existem diversas línguas que nos permitem “falar” com os computadores.

Entre as línguas de produção, o Python é uma das mais fáceis de aprender e uma das que mais cresce no mundo em termos de utilização.

Pode ser utilizado em diversas áreas:

- Data Science;
- Automação de processos;
- Desenvolvimento de sites;
- Inteligência artificial;
- Vários outros

Curiosidade: Seu nome apesar de geralmente ser vinculado a cobra, não tem essa origem.... Na verdade, ele é uma homenagem a um grupo de comédia inglês chamado Monty Python.

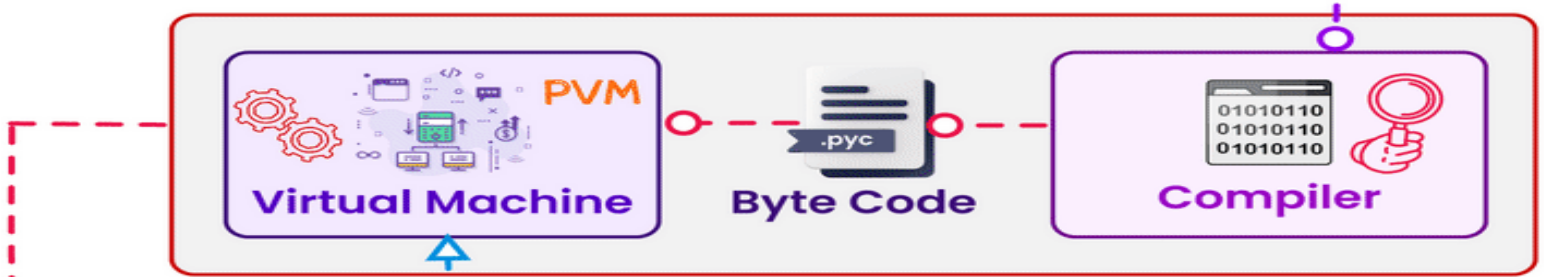
Brij Kishore Pandey

DON'T FORGET TO SAVE

How Python Works

Code Editor

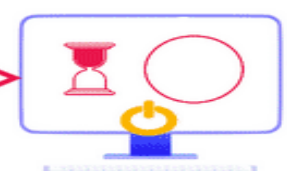
Source File



Python Interpreter

Library Modules

Binary Code / Machine Code



Running Program

Sobre a linguagem

Criada por Guido van Rossum em 1991

Origem do nome: grupo de humoristas *Monty Python*

Linguagem de altíssimo nível (VHLL) Sintaxe simples e fácil de ser assimilada Fácil de usar, aprender, ler

Orientada à objetos, estruturada e funcional

Tipagem forte e dinâmica (Seria aquela em que os objetos/variáveis tem um tipo bem definido e que precisa ser informado no momento de sua declaração.

Interpretada Ambiente interativo

Mais sobre a linguagem

- Extremamente portável (Multiplataforma)
  - Unix/Linux, Windows, Mac, PalmOS, WindowsCE, RiscOS, VxWorks, QNX, OS/2, OS/390, AS/400, PlayStation, Sharp Zaurus, BeOS, VMS...
- Compila para byte code
  - compilação implícita e automática
- Gerenciamento automático memória (Garbage Collector)
- Poderosas estruturas de dados nativas
  - Listas
  - Dicionários
- Licença GPL-compatível

Mais sobre a linguagem

Python é uma linguagem de propósito geral; Não é focada em algo específico;

Simples, fácil e intuitivo;

Multiplataforma (diferentes sistemas operacionais);

Batteries include – ele vem com muitos pacotes instalados;

Código aberto;

Organizada;

Orientada a objetos;

Muitas bibliotecas;

Ainda mais sobre a linguagem

- Tudo é objeto
- Pacotes, módulos, classes, funções
- Tratamento exceções
- Sobrecarga de operadores
- Identação para estrutura de bloco
  - O resto é sintaxe convencional

Mais sobre a linguagem - Resumindo

Python é uma linguagem de propósito geral; Não é focada em algo específico;

Simples, fácil e intuitivo;

Multiplataforma (diferentes sistemas operacionais);

Batteries include – ele vem com muitos pacotes instalados;

Código aberto;

Organizada;

Orientada a objetos;

Muitas bibliotecas;

Sobre a linguagem

https://www.cwi.nl/

Amsterdã - 1982

CWI – Centro de Matemática e Informática

Criadas várias linguagens – ALGOL e ABC

1989 ao ter problema com a linguagem C veio a linguagem Python

No CWI tudo que era criado remetia a programas de televisão.

Assim o criador batizou o seu projeto de *Monty Python Flying Circus*

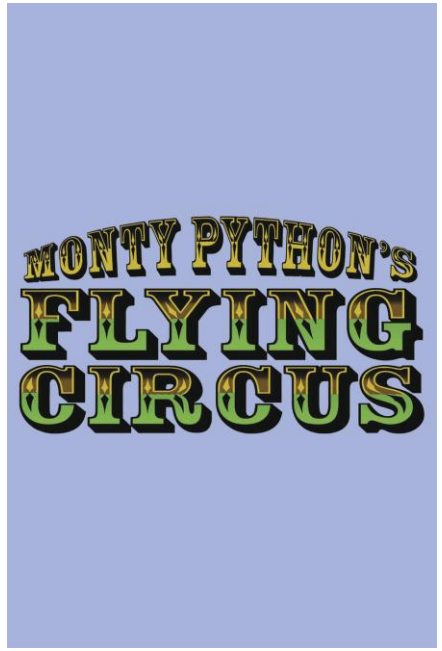


Imagem Monty Python

https://www.youtube.com/watch?v=scD4_ZVDD-8

Sobre a linguagem

Morre Terry Jones, o "menino travesso" do Monty Python

22 JAN 2020 18h18 *atualizado em 23/1/2020 às 07h05*

0 COMENTÁRIOS

Ator, escritor e diretor da icônica e irreverente trupe de comediantes que revolucionou o humor britânico sofria de um raro tipo de demência. Ele era considerado a inspiração para a comédia anárquica do grupo.Terry Jones, um dos fundadores da trupe de comédia anárquica Monty Python, morreu nesta terça-feira (21/01) aos 77 anos de idade. Ele sofria de demência frontotemporal, um distúrbio neurológico raro.

Sobre a linguagem – Mas por que tem duas pythons no símbolo?

Devido a um dos primeiro livros de programação de python que saiu no mercado da editora O’Reilly. Cada livro da editora tem por padrão vir com um animal na capa. Dai vem o símbolo.

O'REILLY®

Sobre a linguagem

Guido saiu da Holanda e veio para os Estados Unidos devido a popularização, onde ele resolveu ensinar a programar, lançando o programa CP4E – **Computer Programming for Everibody, com financiamento da Agência DARPA. Departamento americano responsável pela criação da Internet**

Sobre a linguagem

Em 2001, foi criada a **Python Software Foundation** que mantenedora e coordenadora do python.

Quem patrocina:
Microsoft
Google e Globo.com

Sobre a linguagem

Sistemas operacionais que vem instalado:

Amiga

MacOs

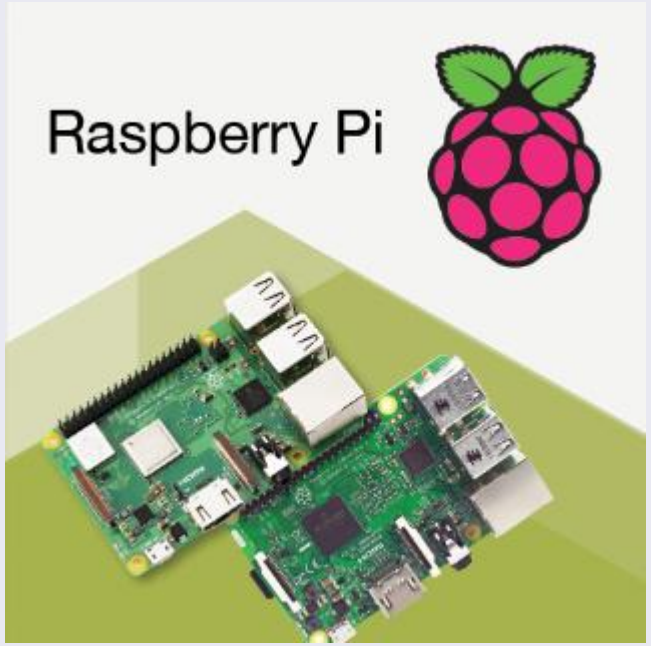
Linux

python SOFTWARE FOUNDATION

Sobre a linguagem

Internet das coisas:

Raspberry Pi



Sobre a linguagem

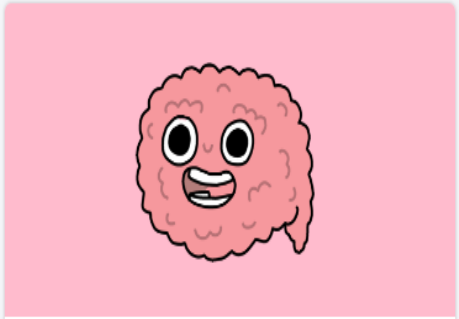
https://projects.raspberrypi.org/pt-BR/projects

Ensino de jogos:

Projetos

Português do Brasil

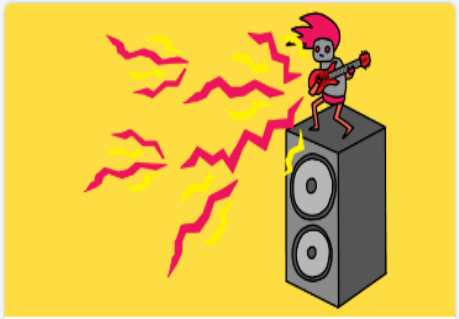
Encontre um projeto : Qualquer tópico | Qualquer hardware | Qualquer software | Qualquer nível



Jogo da Tabuada

Neste projeto você aprederá a construir um jogo com a tabuada, no qual o jogador terá que acertar quantas respostas forem possíveis em 30 segundos.

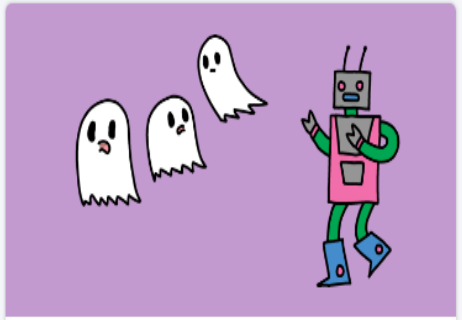
Scratch



Banda de Rock

Aprenda a programar os seus próprios instrumentos musicais

Scratch



Caça-Fantasmas

Crie um jogo de Caça-Fantasmas!

Scratch

E por onde anda o Guido?

ZEN OF PYTHON

- O Zen do Python, por Tim Peters

Bonito é melhor que feio.
Explícito é melhor que implícito.
Simples é melhor que complexo.
Complexo é melhor que complicado.
Linear é melhor do que aninhado.
Esparso é melhor que denso.
Legibilidade conta.
Casos especiais não são especiais o bastante para quebrar as regras.
Ainda que praticidade vença a pureza.
Erros nunca devem passar silenciosamente.
A menos que sejam explicitamente silenciados.
Diante da ambiguidade, recuse a tentação de adivinhar.
Deveria haver um — e preferencialmente só um — modo óbvio para fazer algo.
Embora esse modo possa não ser óbvio a princípio a menos que você seja holandês.
Agora é melhor que nunca.
Embora nunca frequentemente seja melhor que *já*.
Se a implementação é difícil de explicar, é uma má ideia.
Se a implementação é fácil de explicar, pode ser uma boa ideia.
Namespaces são uma grande ideia — vamos ter mais dessas!

Principais áreas

- Inteligência Artificial;
- Biotecnologia.
- Computação 3D

https://www.facebook.com/PlayGroundBR/videos/minha-grande-amiga-gu-gu/590130631381919/

Quem usa Python?

Nasa, Youtube, Zope, Google

Battlefield

Frets on fire

Por que usar Python?

- Uma das linguagens mais divertidas que se tem atualmente
- Já vem com “baterias inclusas” (vasto repertório de bibliotecas)
- Protótipos rápidos sem preocupação com detalhes de implementação da linguagem
  - Linguagem Interpretada: evita “codifica-compila-roda”
- Bem menos linhas de código comparando com Java, C/C++...

CARREIRA

5 linguagens de programação promissoras para se estudar em 2020

Marylene Guedes 10 De Dezembro De 2019

Dart


Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(
title: Text(widget.title),
),
body: Column(
children: <Widget>[
HeroImage(),
...todaysDiscounts,
for (var d in items) ItemWidget(
],
),
);
}
Paint your UI to life
with Dart VM's instant hot reload

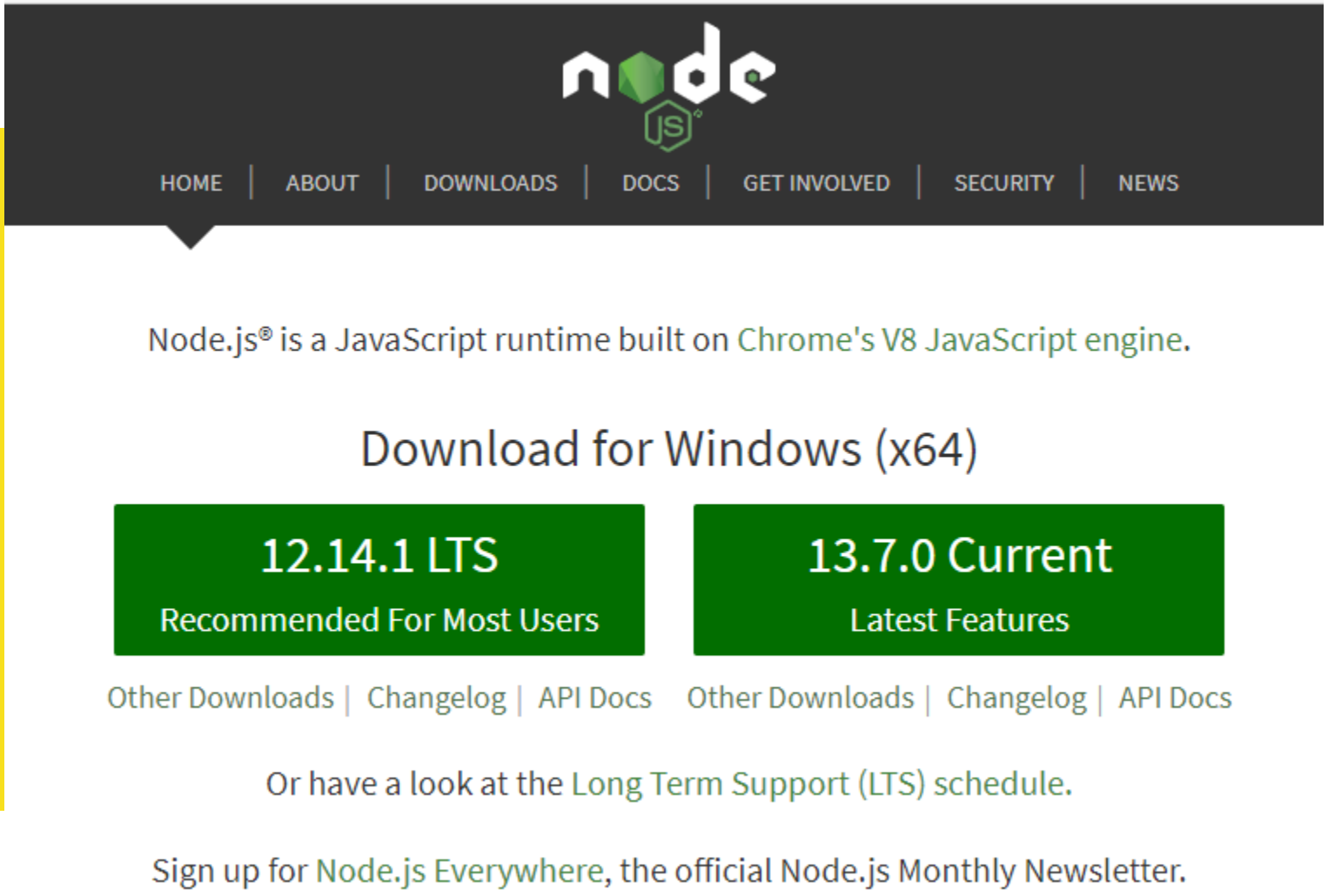
- O Dart tem ficado cada vez mais em evidência, principalmente por causa do Flutter. Trata-se de uma linguagem de programação fortemente tipada criada pela Google em 2011, sendo uma linguagem que pode ser executada tanto em ambientes de aplicações (como em aplicações mobile e desktop) como em ambientes web (podendo até mesmo ser utilizado para o desenvolvimento de aplicações com o Angular).

Kotlin


Kotlin
LEARN
COMMUNITY
TRY ONLINE
Kotlin Census
Your feedback is important
Kotlin 1.3 Released
Kotlin/Native
reached beta
Coroutines
are now stable
Multiplatform
code reuse
Learning Kits
start easily
Programming language for
JVM
Android
JS
Browser
10110
01010
10101
Native

- O Kotlin é uma linguagem de programação open source, multiplataforma e multiparadigma; tendo sido criada em 2010 pela JetBrains. Apesar o Kotlin ter mais apelo na comunidade pelo seu suporte para o desenvolvimento de aplicações Android, ele também pode ser utilizado no desenvolvimento backend.
- Utilizando o Kotlin também para o desenvolvimento backend, é possível obter toda a extensibilidade e maturidade característica dos frameworks e bibliotecas Java através de uma linguagem muito menos verbosa e com uma API mais agradável e moderna.

JavaScript / Node.js



- O Node.js é um ambiente de execução Javascript baseado na engine V8 do projeto Chromium. Ele foi o grande responsável por popularizar o JavaScript como ferramenta para desenvolvimento de aplicações server-side. Outros conceitos utilizados por várias outras ferramentas, como o event looping e a abordagem reativa e não bloqueante, foram inicialmente popularizados pelo Node.js.
- Atualmente, o Node.js é utilizado de maneira maciça por muitas grandes empresas, como Netflix, PayPal e LinkedIn.

Go

A linguagem de programação Go

Documentação | O Projeto | Ajuda | Blog

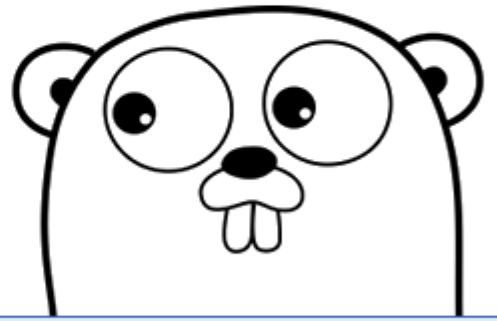
Experimente Go

Pop-out

```
// Você pode editar este código!
// Clique aqui e comece a digitar.
package main

import "fmt"

func main() {
        fmt.Println("Olá, 世界")
}
```

Olá, Mundo! | Execute | Compartilhe | Tour

Go é um ambiente de programação de código aberto que faz com que seja fácil de construir software simples, confiável e eficiente.



Baixando Go

Binários estão disponíveis para Linux, Mac OS X, Windows, e mais.

- Go é uma linguagem de programação escalável e multiplataforma criada pela Google. É fortemente tipada, segura e tem uma biblioteca padrão com um suporte nativo muito rico à grande parte das operações essenciais hoje em dia, como operações de I/O, de manipulação de *streams* e paralelização. Porém, mesmo com todo este poderio, o Go é relativamente fácil de aprender, graças a suas APIs simples aliadas a uma sintaxe concisa e com poucos ruídos.

Por que usar Python?

- O Python é uma linguagem de programação simples e extremamente poderosa. Se você é iniciante, o Python pode ser uma boa escolha, pois é uma linguagem com uma curva de aprendizagem baixa, possibilitando a criação de aplicações em um curto tempo de estudo. Essa característica tem feito com que o Python seja largamente empregado em áreas que atualmente estão em grande expansão no mercado, como *Data Science* e *Machine Learning*. Por causa da crescente demanda por profissionais nas áreas de Data Science e *Machine Learning*, além da presença constante do Python nos ranking de linguagens mais amadas pela comunidade nos últimos anos, estudar Python pode ser uma excelente escolha para 2020.

Por que usar Python?

Python: principal caso de uso é para ciência de dados, mostra estudo

A JetBrains, criadora da plataforma PyCharm IDE for Python, divulgou os resultados da pesquisa Python Developers 2018, um retrato de ferramentas, preferências e percepções de mais de 20 mil desenvolvedores de Python em todo o mundo.

A pesquisa, que ouviu profissionais de mais de 150 países - incluindo o Brasil -, mostra que o uso geral do Python está crescendo, com a análise de dados emergindo como o principal caso de uso, embora desenvolvimento, teste e automação da web ainda estejam em alta.

Por que usar Python?

Python: principal caso de uso é para ciência de dados, mostra estudo

Os conjuntos de tarefas aos quais o Python está associado desde a sua criação ainda estão bem representados: automação de sistema (43%), web scraping (37%), teste de software (32%), todos ainda figuram fortemente.

QUAL LINGUAGEM DE PROGRAMAÇÃO

APRENDER PRIMEIRO?

O QUE É PROGRAMAR?

É escrever instruções bem específicas para uma máquina totalmente idiota, mas muito obediente.

LINGUAGENS

- PYTHON
- JAVA
- C
- PHP
- C++
- JAVASCRIPT
- C#
- RUBY
- OBJECTIVE-C

INICIO

POR QUE VOCÊ QUER APRENDER A PROGRAMAR?

Não sei, apenas escolha uma pra mim

Para meus filhos

Começe com Scratch, depois siga com...

Por diversão

Porque sim

Mais conhecimentos

Tem uma ótima ideia na cabeça?

Não. Só quero começar logo

Prefiro aprender do jeito...

Mais fácil

Melhor

Um pouco difícil

Carro automático ou manual?

Automático

Manual

Da mais difícil (mas que seja fácil aprender outras linguagens depois)

SIM!

Qual plataforma/área?

Conseguir emprego

Ganhar dinheiro

Tenho uma ideia para startup!

Qual plataforma/área?

Quero trabalhar para as grandes empresas de tecnologia!

Tanto faz, só quero $$$

Front-end (interface web)

Web

Back-end ("por trás" do website)

Quero trabalhar para...

Startup

3D/Jogos

Mobile

Qual sistema?

iOS

Android

Corporativo

Empresas

O que você acha da Microsoft?

Sou fã!

Não ligo

Detesto!

Web

Sua aplicação fornecerá dados em tempo real, tipo twitter?

SIM

NÃO

Você quer fazer algo com muito pontecial, mas não tão complexo?

NÃO SEI

NÃO

Qual seu brinquedo favorito?

Lego

Massinha

Ferrorama

Programming is an Art