

```
1  
2  
3 Workshop Python Basic {  
4  
5     [Aula 1]  
6  
7         print("Visão geral e +")  
8  
9  
10  
11  
12     }  
13  
14 }
```



1

2

3

4

5

6

7

8

9

10

11

12

13

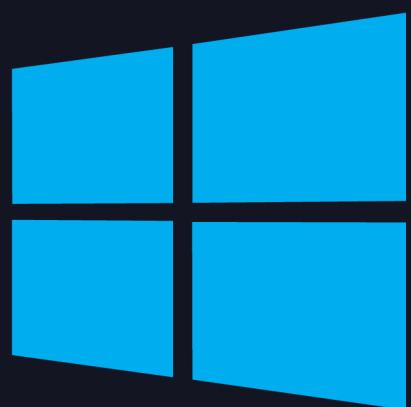
14

01 {

[O que é hardware e software]

Apresentação do conceito de
software e exemplos.

}



1 O que é hardware? {

2 É a parte física de um dispositivo
3 eletrônico.

4 }

5 Dispositivos {

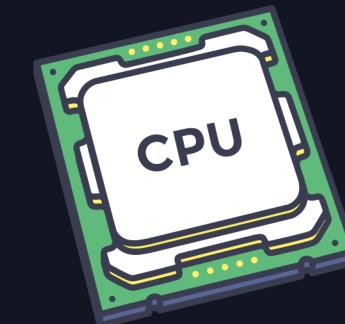
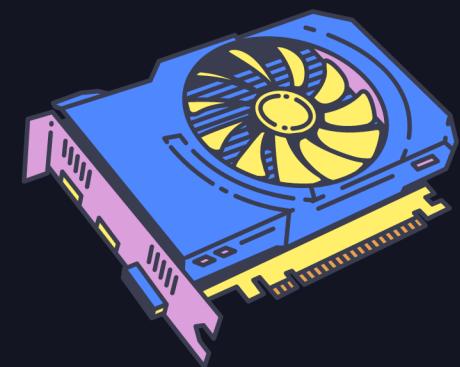
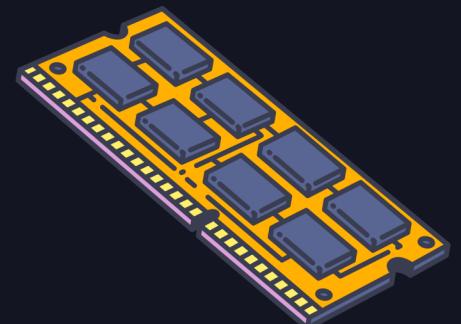
6 Entrada: mouse, teclado, microfone e controle.

7 Armazenamento: Memória Principal, Pen Drive.

8 Processamento: CPU, GPU, ULA e unidade de controle.

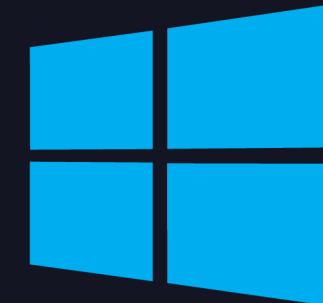
9 Saída: monitor, fone e impressora.

10 }



0 que é software? {

Conjunto de instruções, dados ou programas que controlam o funcionamento de qualquer dispositivo eletrônico.



}

Exemplos {

Sistemas Operacionais (Windows, Linux, MacOS), navegadores (Google, Opera), editores de texto (Word) e jogos (CS:GO, Minecraft).



}

1
2
3
4
5
6
7
8
9
10
11
12
13
14

02 {

[Algoritmos]

Conceito, exemplos e o
pensamento computacional.

}



1 0 que é um algoritmo? {
2
3
4

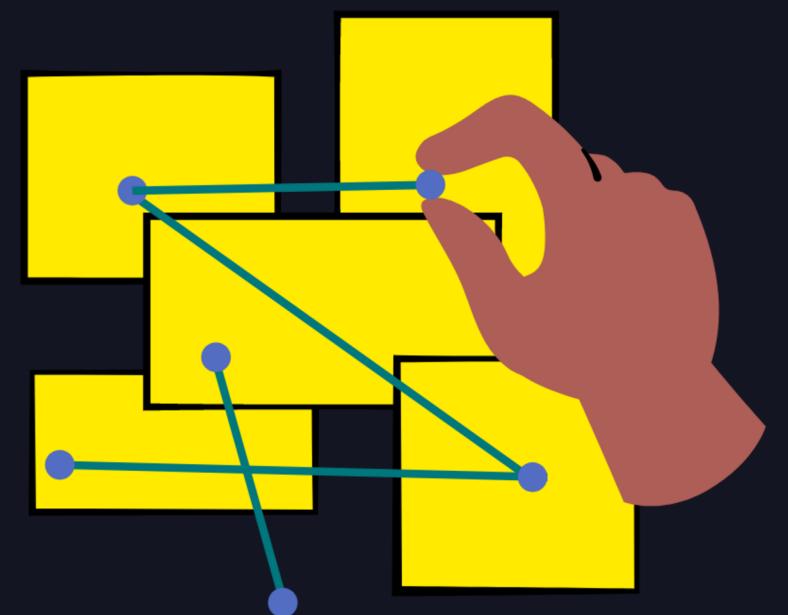
Conjunto de regras e operações bem definidas para resolver um ou mais problemas em um número finito de etapas.

5
6 }
7

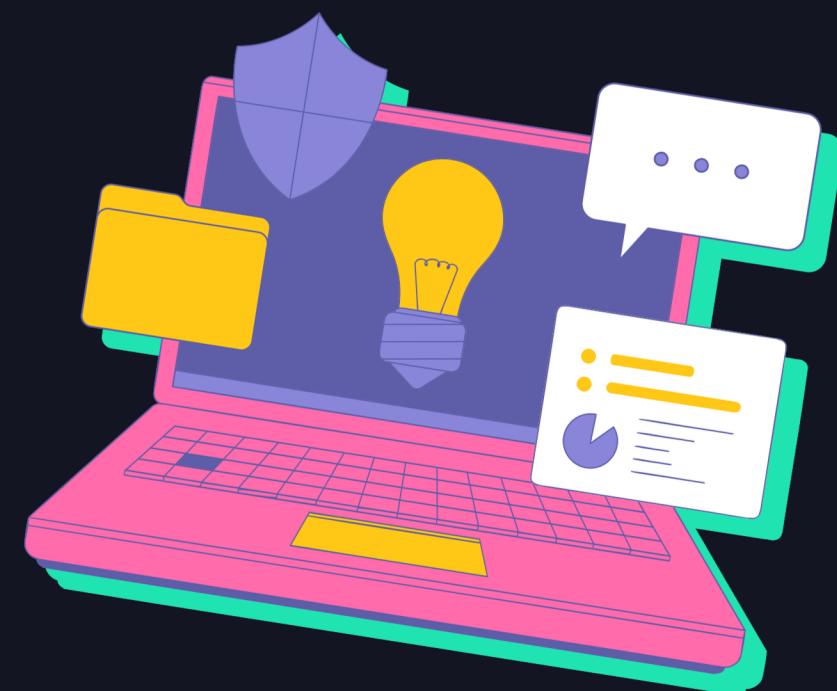
8 0 que é um programa? {
9

10 Sequência completa de instruções a serem
11 executadas por um computador de acordo com um
12 algoritmo.

13 }
14 }



1 Características de um algoritmo {
2
3
4 01 - Clareza e definição
5
6 02 - Efetividade
7
8 03 - Finitude
9
10 04 - Entrada e saída
11
12 }
13
14 }



1 Exemplo {
2

3 Algoritmo em pseudocódigo para fritar um ovo:

4. Pegar o ovo, a frigideira, óleo e sal
5. Colocar o óleo na frigideira
6. Acender o fogo
7. Colocar a frigideira no fogo
8. Esperar o óleo esquentar
9. Quebrar e colocar o ovo na frigideira
10. Colocar o sal a gosto
11. Se estiver pronto, retirar o ovo
12. Se não, esperar mais um pouco
13. Desligar o fogo

14 }



Pensamento computacional {

3 Decomposição: dividir o problema principal em
4 subproblemas para melhor gerenciabilidade.

6 Reconhecimento de padrões: procurar por padrões
7 visuais ou semelhanças com problemas já resolvidos.

10 Abstração: focar no essencial, ignorando detalhes
11 irrelevantes para a solução.

13 }
14 }

Exercício relâmpago! {

Crie um algoritmo em pseudocódigo que descreva o caminho da sua casa até você pegar um ônibus na parada.

}



1
2
3
4
5
6
7
8
9
10
11
12
13
14

03 {

[Variáveis]

Conceito, tipos e como
utilizá-las.

}



O que são variáveis? {

Elementos que permitem que o programador guarde e represente informações. Como se fosse uma caixinha que pode guardar vários tipos de informação. Ou um copo que pode armazenar materiais diferentes.

Para declarar uma variável em python, seguimos a seguinte estrutura:

sinal de atribuição

cidade = “Belém” — valor que queremos atribuir à variável

13 }
14 } Em Python, não é necessário declarar o tipo da variável.

Tipagem de dados {

1 Tipagem estática: a linguagem não permite que o
2 programador altere o tipo de uma variável ao longo da
3 execução de um programa.

4
5 Tipagem dinâmica: a linguagem permite que o
6 programador altere o tipo de uma variável ao longo
7 da execução de um programa.

8
9 Tipagem forte: a linguagem realiza a inferência de
10 tipos automaticamente.

11 Tipagem fraca: a linguagem não realiza a inferência de
12 tipos automaticamente. Python é uma linguagem de tipagem
13 dinâmica e forte.

14 }

```
1 Declaração de variável em Java {  
2  
3     tipo da      — float meia = 2.3      — valor que queremos  
4     variável           |                   atribuir à variável  
5  
6             nome da variável  
7 }  
8
```

```
9 Declaração de variável em C++ {  
10  
11    tipo da      — string txt = "oi"      — valor que queremos  
12    variável           |                   atribuir à variável  
13  
14 }
```

```
1  Tipos de variáveis {  
2  
3      |   Inteiro (int): Valores inteiros (3, -10, 28, 150)  
4  
5      |   Ponto flutuante (float): Números decimais (3.14, 2.5, 1.74)  
6  
7      |   String (str): Sequências de caracteres ("palavras",  
8                           "frases", "10", "3.1415")  
9  
10     |   Booleano (bool): Valores lógicos:  
11                           verdadeiro (True) ou falso (False)  
12  
13 }  
14 }
```

Nomeando variáveis {

- Nomes claros e objetivos.
- Não pode usar caracteres especiais além do underline "_".
- Não pode começar com número.
- O nome de variáveis é case sensitive
 - aulanumer01 X aulaNúmero01
- Não pode conter espaços.
- Busque sempre padronizar a forma que você nomeia suas variáveis.
 - camelCase
 - snake_case
 - PascalCase
- Existem palavras reservadas

}

Exercício relâmpago! {

Crie variáveis e atribua os seguintes valores:

→ inteiro;

→ ponto flutuante;

→ string;

→ bool;

Obs: Use uma convenção de nomenclatura
(camelcase, snakecase, ...)

}



1
2
3
4
5
6
7
8
9
10
11
12
13
14

04 {

[Entrada e saída de
dados]

O que são e como funcionam?

}

```
1 Entrada de dados {  
2 |  
3 |   função input()  
4 |  
5 |       nome = input("Qual é o seu nome? ")  
6 |           |  
7 |           variável que queremos  
8 |           | atribuir o valor  
9 }  
10 Saída de dados {  
11 |  
12 |   função print()  
13 |  
14 }  
|  
|   print("Hello, world!")  
|  
|       |  
|       dado que  
|       | queremos exibir  
|  
|       |  
|       função para saída  
|       | de dados
```

Tipos de variável no input{

- Todo dado recebido por um input é uma string;
- Podemos “escolher” o tipo da variável que queremos receber.

```
idade1 = input("Qual é a sua idade? ")          idade2 = int(input("Qual sua idade? "))

> 20                                         > 23.4

(tipo: string - "20")                         (tipo: int - 23)

}
```

Função print() - f-strings {

Permitem que variáveis sejam formatadas dentro da mensagem do `print()`. Ajuda na organização da saída do código.

Basta colocar a letra “f” antes do texto e o nome da variável entre chaves {}.

Exemplo:

```
nome = "Maria"  
idade = 18  
print(f"{nome} tem {idade} anos.")
```

Saída: Maria tem 18 anos.

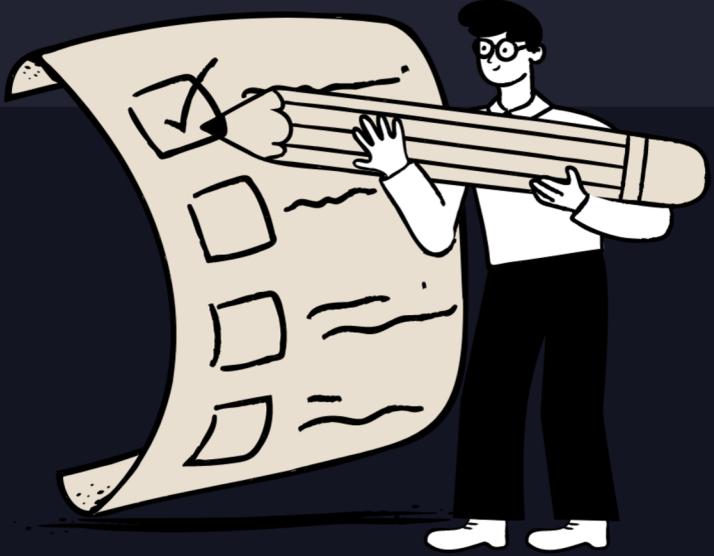
1
2
3
4
5
6
7
8
9
10
11
12
13
14

04.1 {

[Hora de praticar!]

Vamos fazer um exercício?

}

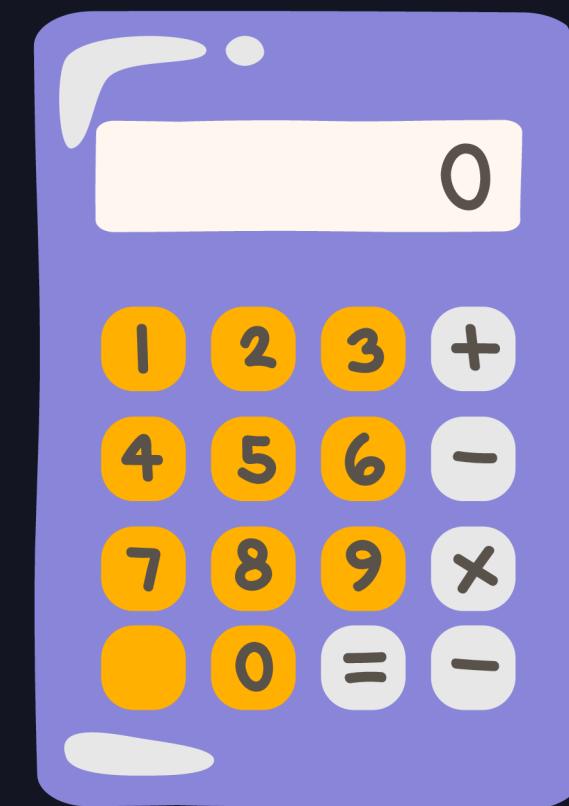


1
2
3
4
5
6
7
8
9
10
11
12
13
14

05 {

[Operadores e Estruturas]

Aritméticos, relacionais,
lógicos e condicionais.



}

Operadores aritméticos {

Utilizados na execução de operações matemáticas.

Soma (+)

Subtração (-)

Multiplicação (*)

Exponenciação (**)

Divisão (/)

Divisão inteira (//)

Módulo da divisão (%)

Qual a diferença?

$10 / 3 = 3.333$

$10 // 3 = 3$

$10 \% 3 = 1$

}

```
1 Operadores relacionais {  
2  
3     Utilizados para comparar dois valores.  
4     > - maior que;  
5     < - menor que;  
6     >= - maior ou igual a;  
7     <= - menor ou igual a;  
8     == - igual a;  
9     != - diferente de;  
10  
11 }  
12  
13  
14 }
```

True ou False?
1 < 2 ?
3 >= 3 ?
7 != 19 ?
“oi” == “OI” ?

Operadores lógicos{

Utilizados para criar expressões lógicas.

AND (E) - Retorna verdadeiro caso todas as expressões sejam verdadeiras.

OR (OU) - Retorna verdadeiro caso pelo menos uma das duas expressões seja verdadeira.

NOT (NÃO) - Retorna o oposto do valor da expressão.

}

Estruturas condicionais: {

São expressões lógicas ou relacionais que podem ser satisfeitas ou não.

Exemplos:

Se você tem 18 anos de idade ou mais,
então você pode tirar sua carteira de habilitação.

}

Satisfazer uma condição significa que ela recebe o valor “True”.

Estrutura “if-elif-else” {

3 Utilizada para avaliar mais de uma condição. “**elif**”
4 é uma abreviação de “**else if**” (senão, se)

```
5     if(hora >= 6 and hora < 12):
6         print("Bom dia!!")
7
8     elif(hora >= 12 and hora < 18):
9         print("Está de tarde!")
10    elif(hora >= 18 and hora < 0):
11        print("Boa noite!")
12    else:
13        print("Vai dormir!!")
14 }
```

Nota :

Podemos utilizar tanto operadores relacionais quanto operadores lógicos para estabelecer condições.

Indentação {

É o **espaço** equivalente a um “**tab**” que indica quais linhas de código estão “**aninhadas**” à condição. É o **escopo** de um comando.

```
if(passagem >= 2.30):
    print("Você paga meia-passagem.")
    print("Você pode pagar a passagem inteira.")
    print("Você consegue andar de ônibus em dias de semana.")

else:
    print("Você não consegue andar de ônibus em dias de semana.")
```

Extremamente importante para a organização do código em estruturas de condição, repetição, definição de funções e etc.

1
2
3
4
5
6
7
8
9
10
11
12
13
14

05.1 {

[Hora de praticar!]

Praticando que se aprende!

}

