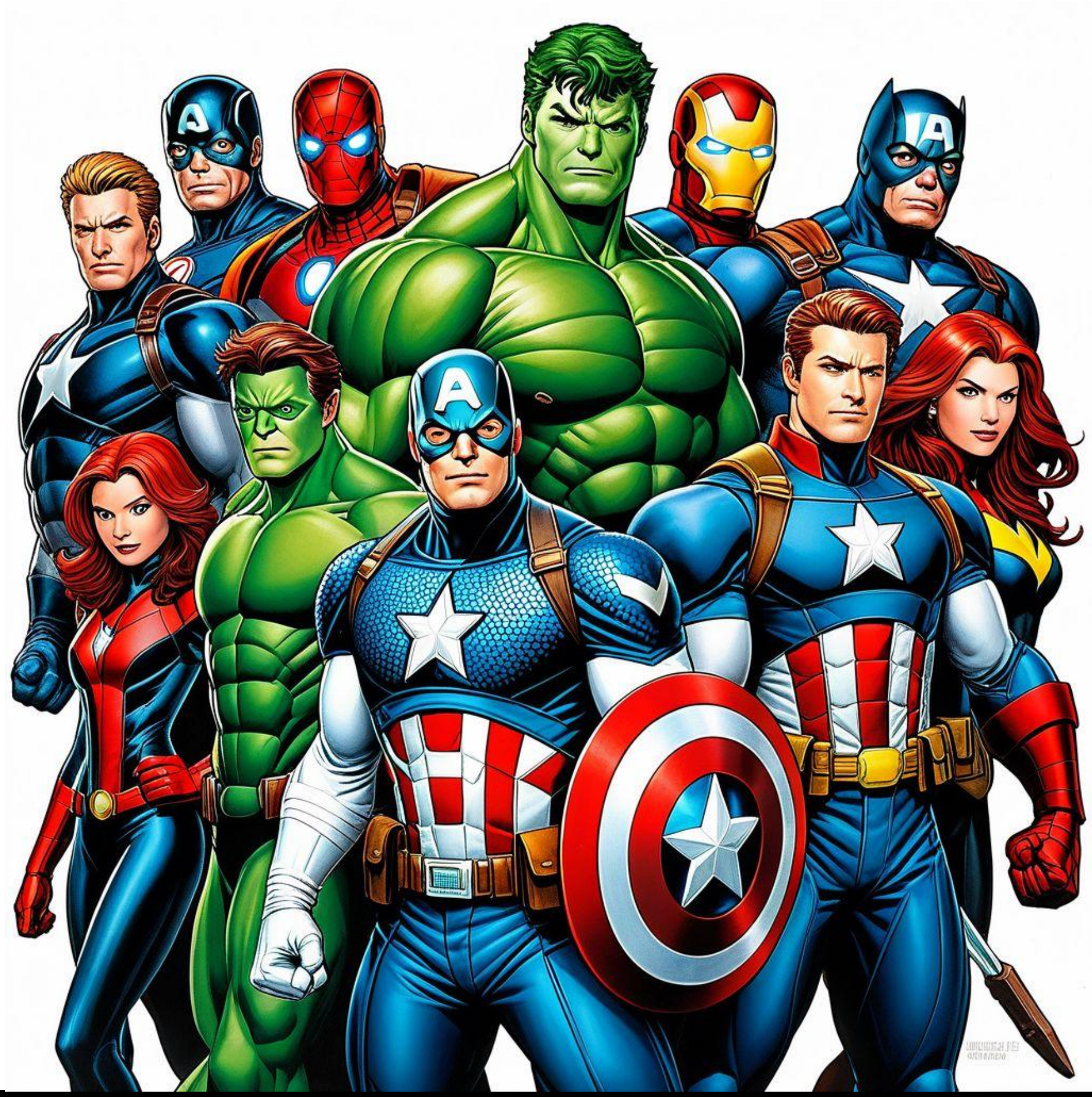
A Liga Python

DESVENDANDO OS 10 PRINCIPAIS COMANDOS

PARA INICIANTES



BIANCA MARTINS

O Começo da Sua Jornada em Python

Aprenda os 10 Fundamentos Essenciais

Se você está começando a sua jornada no mundo da programação, Python é uma das melhores escolhas que você pode fazer. Ela é uma linguagem simples, poderosa e amplamente utilizada em diversas áreas, como desenvolvimento web, automação, análise de dados e até inteligência artificial.

Neste ebook, vamos explorar os 10 comandos mais importantes para quem está iniciando com Python.

A proposta deste material é ser direto e prático, com explicações claras e exemplos simples que você pode aplicar imediatamente. Cada comando será apresentado com um exemplo de código em um contexto real, para que você possa entender não apenas a teoria, mas também como utilizá-los no seu dia a dia de programação.

Com o foco em iniciantes, o objetivo é tornar seu aprendizado mais eficiente e divertido. Ao final deste ebook, você terá as ferramentas necessárias para dar os primeiros passos e começar a criar seus próprios programas em Python.

Vamos lá? Prepare-se para aprender os conceitos fundamentais que vão te levar a um novo patamar na programação!

01

EXIBINDO INFORMAÇÕES COM PRINT( )

Aprenda a usar o comando print() para exibir informações na tela e facilitar a interação e depuração do seu código.

Exibindo Informações com print( )

O comando **print()** é o primeiro passo para interagir com o usuário, exibindo informações na tela.

A LIGA PYTHON - BIANCA MARTINS.PY

```
nome = "João"
idade = 25
print("Nome:", nome)
print("Idade:", idade)
```

Sugestão: Use print() sempre que quiser verificar o valor de uma variável ou mostrar resultados ao usuário. Essencial para testar o código enquanto aprende!

02

INTERAGINDO COM O USUÁRIO: USANDO INPUT( )

Entenda como usar input() para coletar dados do usuário e tornar seu programa mais dinâmico e interativo.

Interagindo com o Usuário: Usando input()

Com **input()**, é possível pedir informações ao usuário. O que ele digitar será retornado como uma string.

A LIGA PYTHON - BIANCA MARTINS.PY

```
nome = input("Qual é o seu nome? ")
print("Olá, " + nome + "!")
```

Sugestão: Use input() quando precisar coletar dados do usuário, como nome, idade ou preferências, para personalizar a interação no seu programa.

03

CONVERTENDO TIPOS DE DADOS: INT( ) E FLOAT( )

Saiba como usar int() e float() para converter strings em números, facilitando operações matemáticas.

Convertendo Tipos de Dados: int() e float()

Em Python, podemos usar **int()** para converter valores em números inteiros e **float()** para valores de ponto flutuante (números decimais).

A LIGA PYTHON - BIANCA MARTINS.PY

```
numero_inteiro = int("10")
numero_float = float("10.5")
print(numero_inteiro + numero_float)
```

Sugestão: Sempre que receber dados como texto, use int() ou float() para garantir que as variáveis possam ser manipuladas como números em cálculos.

04

TOMANDO DECISÕES: IF, ELIF E ELSE

Aprenda a tomar decisões no seu código com if, elif e else, controlando o fluxo do programa.

Tomando Decisões: Estruturas Condicionais com if, elif e else

Essas estruturas permitem que você tome decisões no seu código, executando ações diferentes dependendo das condições.

A LIGA PYTHON - BIANCA MARTINS.PY

```python
idade = 20
if idade >= 18:
    print("Você é maior de idade!")
else:
    print("Você é menor de idade!")
```

Sugestão: Use condicionais quando precisar realizar ações baseadas em diferentes situações, como verificar permissões ou escolher entre alternativas.

05

REPETINDO AÇÕES: COMLAÇOS DE REPETIÇÃO FOR

Descubra como usar o laço for para repetir ações de forma simples, percorrendo listas ou sequências.

Repetindo Ações: Laços de Repetição com for

O laço **for** é usado para repetir um bloco de código para cada item de uma lista ou sequência.

A LIGA PYTHON - BIANCA MARTINS.PY

```
nomes = ["Ana", "João", "Maria"]
for nome in nomes:
    print(nome)
```

Sugestão: Use for para percorrer listas ou outras sequências, como números ou strings, facilitando tarefas repetitivas.

06

REPETINDO ATÉ:

LAÇOS DE REPETIÇÃO WHILE

Veja como usar while para repetir ações até que uma condição seja atendida, útil em loops indefinidos.

Repetindo Até: Laços de Repetição com while

Com **while**, o código será repetido enquanto uma condição for verdadeira, tornando-o útil para situações em que o número de repetições não é conhecido.

A LIGA PYTHON - BIANCA MARTINS.PY

```python
contador = 0
while contador < 5:
    print(contador)
    contador += 1
```

Sugestão: Use while quando precisar que seu programa continue executando enquanto uma condição se mantiver, como em jogos ou em processos de espera.

07

CRIANDO SEQUÊNCIAS DE NÚMEROS COM RANGE( )

Aprenda a gerar sequências de números com range(), ideal para usar em laços de repetição.

Criando Sequências de Números com range()

range() é uma função que cria uma sequência de números, ideal para usar dentro de laços for.

A LIGA PYTHON - BIANCA MARTINS.PY

```
for i in range(5):
    print(i)
```

Sugestão: Utilize range() quando precisar gerar uma sequência de números, por exemplo, para iterar em uma quantidade específica de vezes.

08

DESCOBRINDO O TAMANHO COM LEN( )

Use len() para obter o tamanho de listas, strings e outros objetos iteráveis no seu código.

Descobrindo o Tamanho com len()

len() é utilizado para descobrir o tamanho de uma lista, string ou outro objeto iterável.

A LIGA PYTHON - BIANCA MARTINS.PY

```
frutas = ["maçã", "banana", "laranja"]
print(len(frutas))
```

Sugestão: Use len() sempre que precisar saber quantos itens há em uma lista ou qual o comprimento de uma string.

09

TRABALHANDO COM LISTAS: USANDO LIST( )

Entenda como criar listas a partir de outros objetos com list(), facilitando o manuseio de dados.

Trabalhando com Listas: Usando list()

O comando **list()** converte outras sequências, como tuplas ou strings, em listas, permitindo manipulação fácil de dados.

A LIGA PYTHON - BIANCA MARTINS.PY

```
numeros = list(range(5))
print(numeros)
```

Sugestão: Use list() sempre que precisar trabalhar com uma sequência de dados de forma mutável, como adicionar ou remover itens.

10

ORGANIZANDO O CÓDIGO COM FUNÇÕES: DEF

Aprenda a criar funções com def para organizar seu código, tornando-o modular e reutilizável.

Organizando o Código com Funções: Definindo com def

Com a palavra-chave **def**, você pode criar funções, agrupando tarefas específicas em blocos de código reutilizáveis.

A LIGA PYTHON - BIANCA MARTINS.PY

```
def saudacao(nome):
    return "Olá, " + nome + "!"

print(saudacao("Carlos"))
```

Sugestão: Crie funções para evitar repetição de código e tornar seu programa mais organizado. Sempre que tiver uma tarefa que se repete, pense em criar uma função.