INSTITUTO FEDERAL
Mato Grosso do Sul

Introdução ao Python

Tutorial Básico

Python é uma linguagem de programação muito versátil, que permite a realização de diversas operações matemáticas de forma simples e rápida. Neste tutorial, abordaremos as principais operações matemáticas em Python.

Operadores matemáticos básicos em Python

Python suporta os seguintes operadores matemáticos básicos:

- Adição (+)
- Subtração (-)
- Multiplicação (*)
- Divisão (/)
- Divisão inteira (//)
- Resto da divisão (%)
- Potenciação (**)

Exemplos de uso desses operadores:

```python
a = 10
b = 3
soma = a + b
subtracao = a - b
multiplicacao = a * b
divisao = a / b
divisao_inteira = a // b
resto_divisao = a % b
potencia = a ** b

print("Soma:", soma)
print("Subtração:", subtracao)
print("Multiplicação:", multiplicacao)
print("Divisão:", divisao)
print("Divisão inteira:", divisao_inteira)
print("Resto da divisão:", resto_divisao)
print("Potenciação:", potencia)
```

(Clique na imagem para ver em tamanho maior)

Funções matemáticas em Python

Python também inclui diversas funções matemáticas integradas, como as funções trigonométricas, logarítmicas e exponenciais. Essas funções são fornecidas pelo módulo `math`, que precisa ser importado antes de ser usado.

Exemplos de uso de funções matemáticas:

```python
import math

x = 2
seno = math.sin(x)
cosseno = math.cos(x)
tangente = math.tan(x)
logaritmo = math.log(x)
exponencial = math.exp(x)

print("Seno:", seno)
print("Cosseno:", cosseno)
print("Tangente:", tangente)
print("Logaritmo:", logaritmo)
print("Exponencial:", exponencial)
```

(Clique na imagem para ver em tamanho maior)

Em Python, uma lista é um tipo de dados que permite armazenar uma coleção de itens em uma única variável. A lista é uma estrutura de dados muito versátil e útil para muitas tarefas diferentes, incluindo a manipulação de dados em programas e o armazenamento de informações em sistemas.

Criando uma lista

Para criar uma lista em Python, usamos colchetes [ ] e separamos cada elemento com uma vírgula. Aqui está um exemplo de uma lista de compras:

```
lista_de_compras = ['maçãs', 'bananas', 'laranjas', 'uvas']
```

Podemos criar uma lista vazia simplesmente usando colchetes sem nenhum elemento dentro:

```
lista_vazia = []
```

Acessando elementos em uma lista

Podemos acessar um elemento específico em uma lista usando seu índice, que é a posição do elemento na lista. Em Python, o índice começa em 0. Por exemplo, para acessar o primeiro elemento da lista_de_compras acima, usamos:

```
primeiro_item = lista_de_compras[0]
```

Podemos acessar o último elemento da lista usando um índice negativo. Por exemplo, para acessar o último item da lista_de_compras acima, usamos:

```
ultimo_item = lista_de_compras[-1]
```

Adicionando e removendo elementos em uma lista

Podemos adicionar um novo elemento a uma lista usando o método `append()`. Por exemplo, para adicionar um item de "morangos" à lista_de_compras, usamos:

```
lista_de_compras.append('morangos')
```

Podemos remover um elemento de uma lista usando o método `remove()`. Por exemplo, para remover o item de "bananas" da lista_de_compras, usamos:

```
lista_de_compras.remove('bananas')
```

Slicing (Fatiamento) de listas

Podemos selecionar uma parte específica de uma lista usando a sintaxe de fatiamento (slicing). Para selecionar os elementos de uma lista de compras do segundo ao quarto elemento, usamos:

```
itens_selecionados = lista_de_compras[1:4]
```

Isso retornará uma nova lista contendo os elementos "bananas", "laranjas" e "uvas".

Compreensão de listas

Em Python, podemos criar uma lista de forma mais concisa usando uma compreensão de lista. Por exemplo, para criar uma lista de todos os números pares de 0 a 10, podemos usar:

```
lista_de_pares = [x for x in range(11) if x % 2 == 0]
```