Revisão

Matéria de Hoje

Matéria Da Aula de Hoje

- Criando Funções
- Escopo de variáveis
- Criando Módulos
- Importando Módulos
- Salvar Dados Em Um Arquivo ".txt"
- Abrir Arquivo ".txt" E Recuperar Dados Salvos
- Adicionar Dados Em Arquivos ".txt" Sem Sobrescrever O Original.

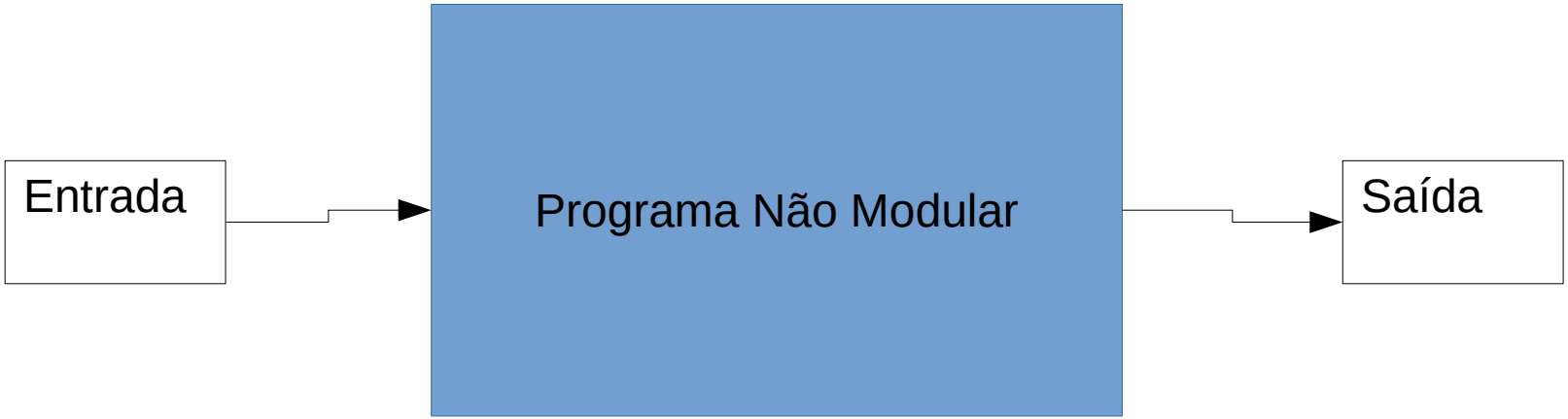
Funções

Criando Funções

Funções

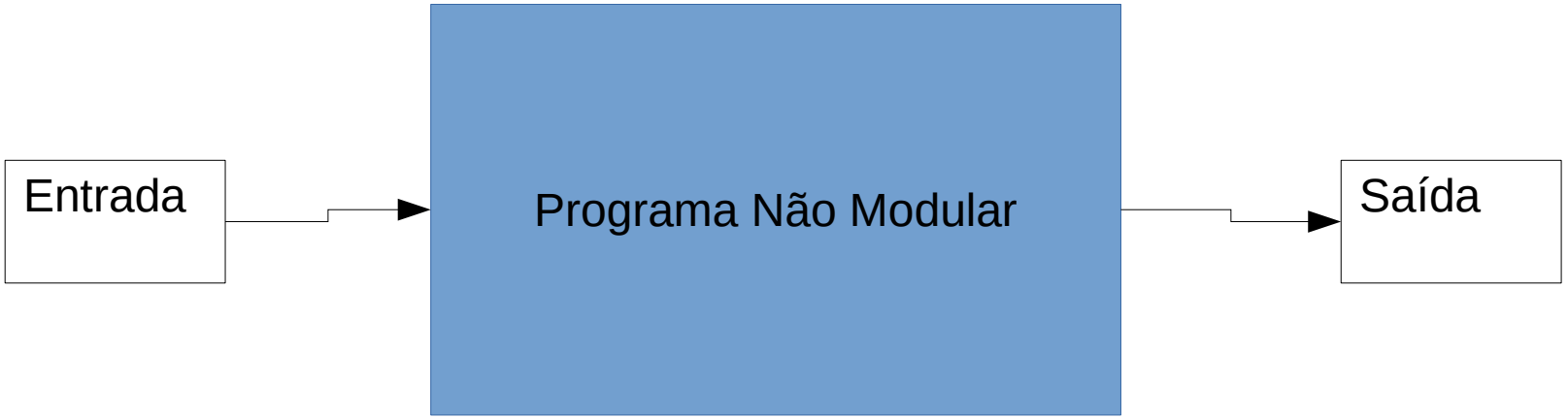
- São objetos do Python formado por uma sequência de comandos.
- A função recebe um nome que é usado para chamá-lo quando necessário.
- Ele ajuda a diminuir a repetição de linhas de código e ajuda na modulação do programa

Funções



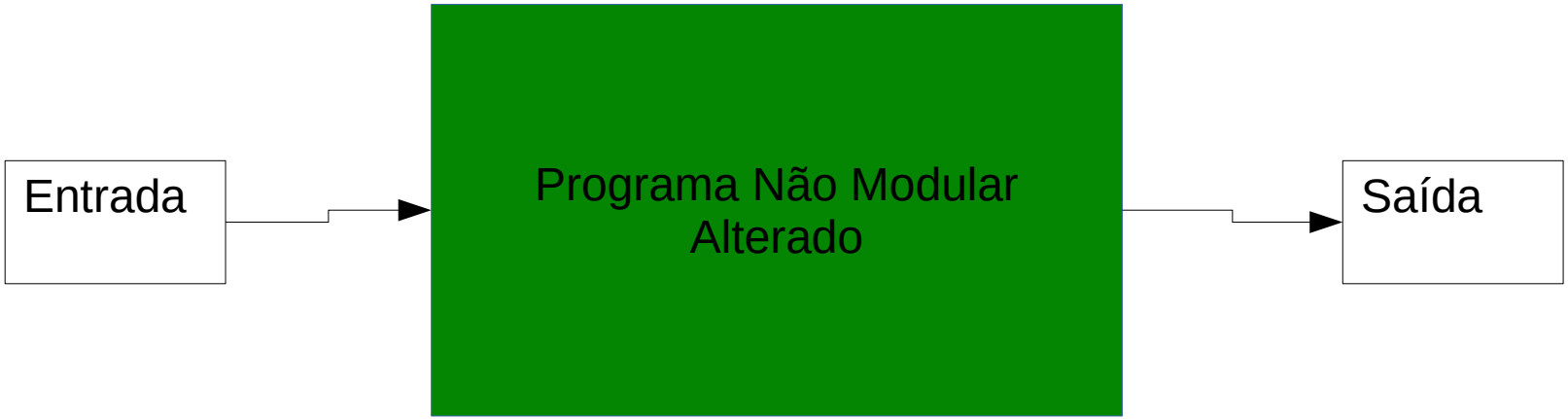
Funções

Para mudar este programa
Você que mudar todo ele



Funções

Para mudar este programa
Você que mudar todo ele

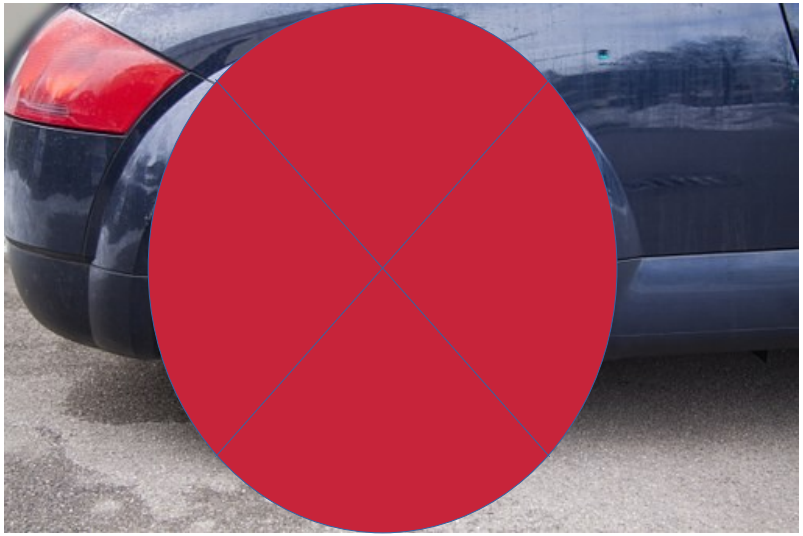


Funções

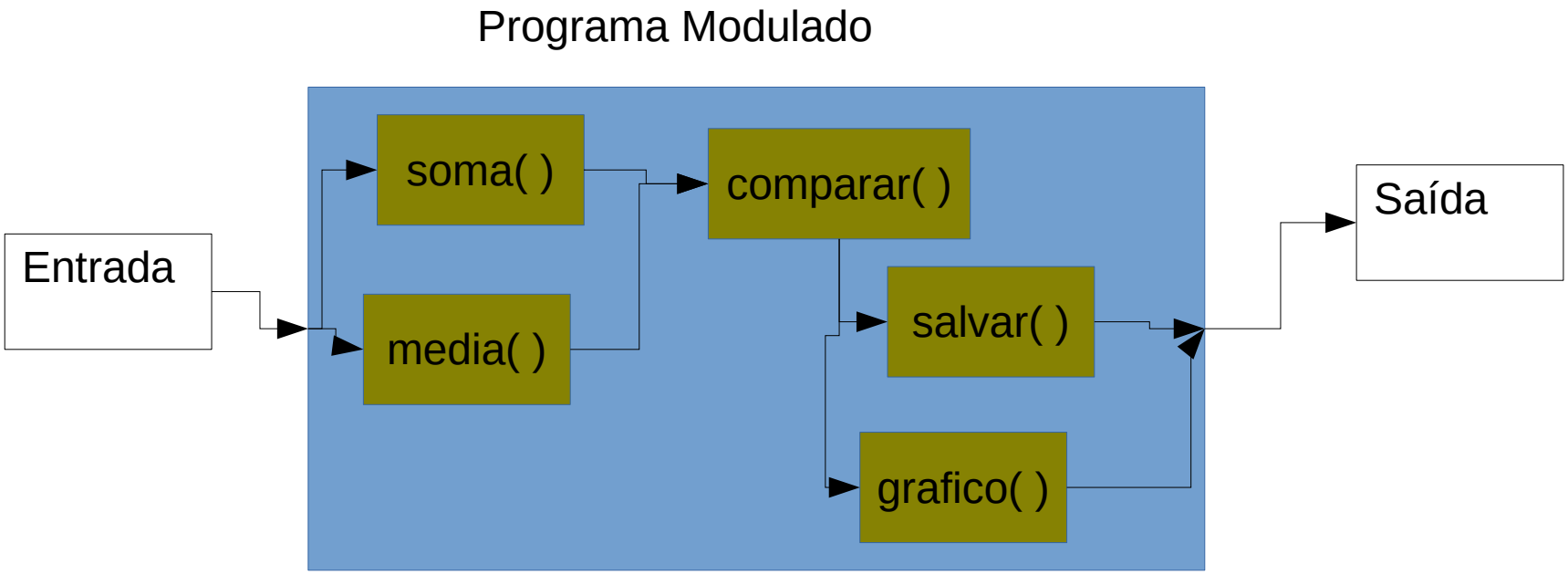
Pneu do carro Furou!

Funções

Solução: Jogo Fora e Compro um novo!



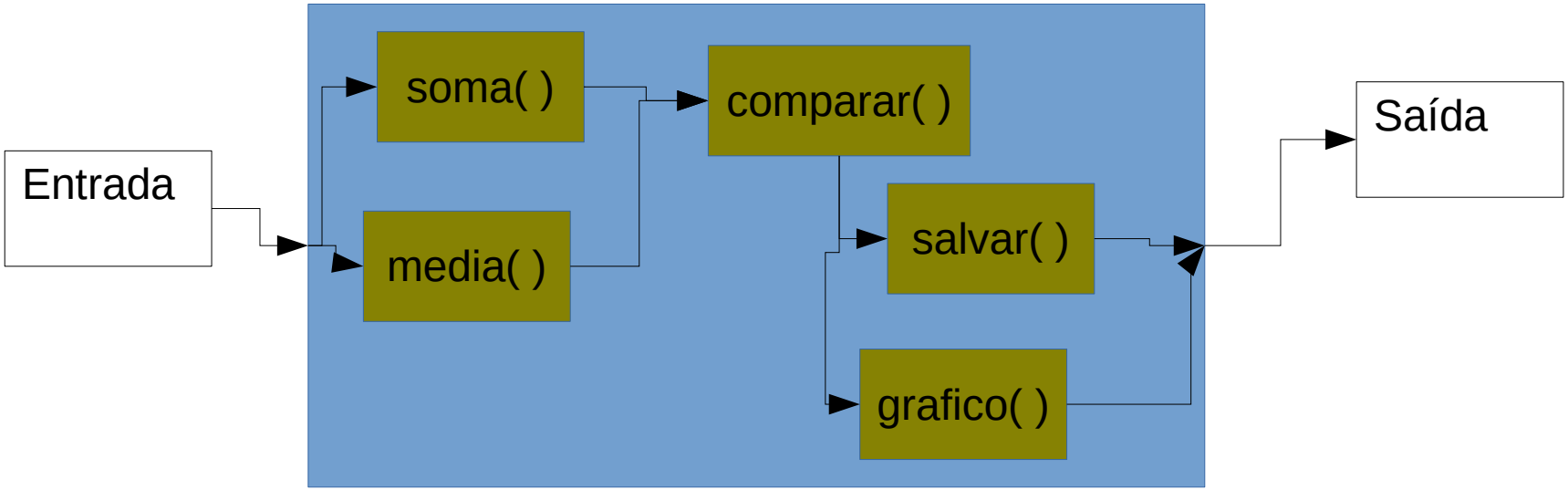
Funções


Programa Modulado
Entrada
soma( )
media( )
comparar( )
salvar( )
grafico( )
Saída

Funções

Alterar e Acrescentar

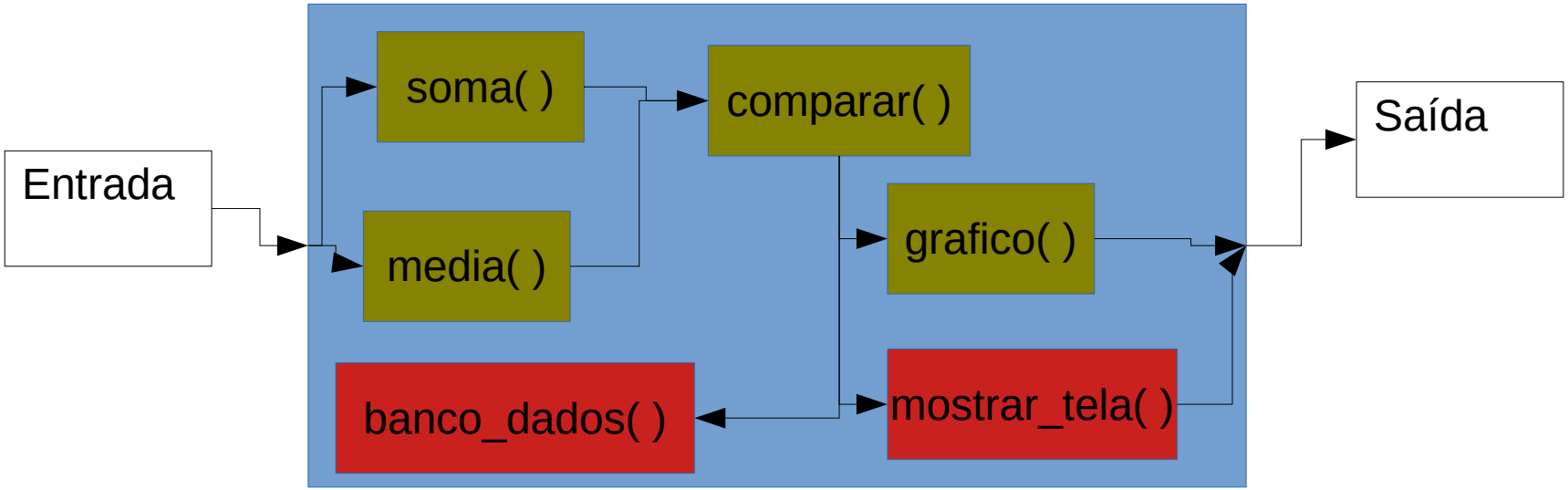
Programa Modulado



Funções

Alterar e Acrescentar

Programa Modulado



Funções

Pneu do carro Furou!

Funções

Conserto o pneu e volto a andar de carro!

Funções

Indentação = 4
espaços em branco

- Criando Funções:

  def nome_funcao(parametros):

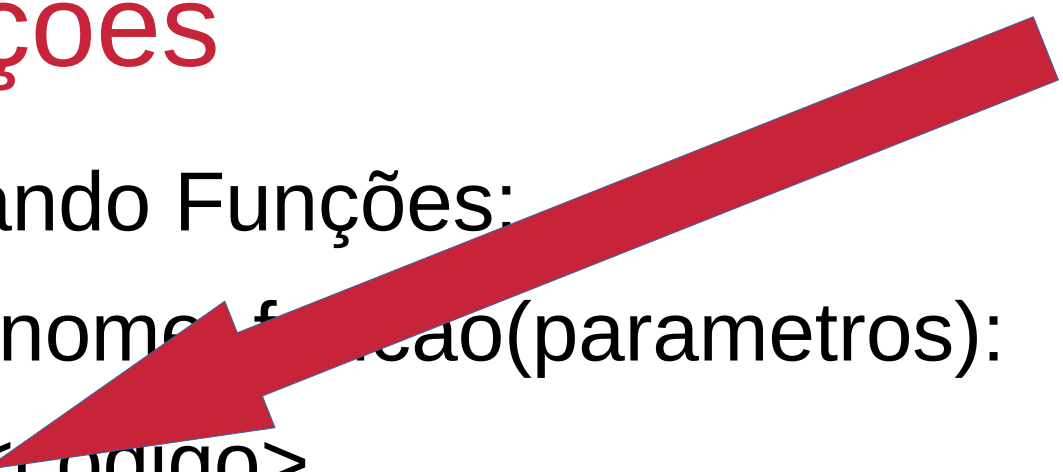
  ….<código>

  ….parametros

  ….return parametros

Funções

Indentação = 4
espaços em branco



- Criando Funções:

  def nome_funcao(parametros):

  ….<código>

  ….parametros

  ….return parametros

Funções

- Exemplo: Soma de dois números

```
def soma (parametro1, parametro2):
    soma_funcao = parametro1 + parametro2
    return soma_funcao
```

Funções

- Para chamar a função soma no código

```
>>> soma( 1, 3)
4
>>> var = 2
>>> var2 = 3
>>> soma(var, var2)
5
```

Escopo de Variáveis

Escopo de Variáveis

```
def soma (parametro1, parametro2):
    soma_funcao = parametro1 + parametro2
    return soma_funcao
```

Escopo de Variáveis

Parâmetros são variáveis que existem somente na função (local). São as portas de entrada para a função.

```
def soma (parametro1, parametro2):
    soma_funcao = parametro1 + parametro2
    return soma_funcao
```

soma_funcao é uma variável local (que existe somente nesta função)

Globais vs Locais

- As variáveis globais existem em todo o programa.
- Não é bom usar variáveis globais dentro das funções

Funções

- Para chamar a função soma no código

```
>>> soma( 1, 3)
4
>>> var = 2
>>> var2 = 3
>>> soma(var, var2)
5
```

As variáveis var e var2 são globais. Estão no código principal e fora de funções e classes.

Note que passamos as variáveis para a função soma.

O Valor da var e var2 será passada para o parametro1 e parametro2

Globais vs Locais

- As variáveis globais existem em todo o programa.
- Não é bom usar variáveis globais dentro das funções
- É recomendado passa-las por parametros

Globais vs Locais

- As variáveis locais só existem no local que são criadas.
- Quando criadas dentro da função, ao terminar a execução, elas serão destruídas

Exercício 11

- 1) Crie uma função que receba 3 notas por parâmetro e retorne a média.
- 2) Crie uma função que receba uma lista de qualquer tamanho e retorne a média
- 3) Crie 4 funções para as operações matemáticas (+, -, /, *):
- 4) Crie um programa que peça 2 números. Depois crie um menu interativo que peça qual operação matemática deseja realizar (+, -, /, *). Utilize as funções criadas no exercício anterior e mostre o resultado da operação escolhida.

Criando Módulos

Criando Módulos

- Umas das vantagens de se criar funções é a possibilidade de retirá-la do código principal e salvá-la em um arquivo separado.
- Isso traz 3 vantagens:
  - Primeira: O código principal fica mais limpo
  - Segunda: Para fazer a manutenção/alteração na função, você abre o arquivo separado sem correr o risco de alterar por acidente o código principal.
  - Terceiro: Você pode usá-la tanto no código principal quanto em outros programas. Tendo assim um reaproveitamento melhor do seu código.

Criando Módulos

- Como fazer:
  - Primeiro: Retira a função do código principal
  - Segundo: Cole a função em um arquivo separado e salve com a extensão .py
  - Terceiro: use o from/import para importar a função.

Criando Módulos

```
#### funções
def soma(par1, par2):
    soma_funcao = par1 + par2
    return soma_funcao

#### Programa principal
num1 = int( input('Digite um número: '))
num2 = int( input('Digite um número: '))
resultado = soma(num1, num2)
print(resultado)
```

Criando Módulos

```
# programa.py
#### funções
def soma(par1, par2):
    soma_funcao = par1 + par2
    return soma_funcao

#### Programa principal
num1 = int( input('Digite um número: '))
num2 = int( input('Digite um número: '))
resultado = soma(num1, num2)
print(resultado)
```

Criando Módulos - Primeiro

```
# programa.py

#removendo a função!

#### Programa principal
num1 = int( input('Digite um número: '))
num2 = int( input('Digite um número: '))
resultado = soma(num1, num2)
print(resultado)
```

Criando Módulos - Segundo

```
# programa.py

#removendo a função!

#### Programa principal
num1 = int( input('Digite um número: '))
num2 = int( input('Digite um número: '))
resultado = soma(num1, num2)
print(resultado)
```

```
# funcao_soma.py

#### funções
def soma(par1, par2):
    soma_funcao = par1 + par2
    return soma_funcao
```

Criando Módulos - Terceito

```
# programa.py

from funcao_soma import soma

#rimportando a função!

#### Programa principal
num1 = int( input('Digite um número: '))
num2 = int( input('Digite um número: '))
resultado = soma(num1, num2)
print(resultado)
```

```
# funcao_soma.py

#### funções
def soma(par1, par2):
    soma_funcao = par1 + par2
    return soma_funcao
```

Importando Módulos

Importando Módulos

- Para importar um módulo usamos o comando:

  from <nome do arquivo> import <nome função>
- No nome do arquivo coloca-se o nome do arquivo que está salvo a função. Nesta parte é interessante colocar o endereço parcial ou total do arquivo.
- O nome da função é o nome da função que se deseja importar. Pode ser uma única função ou mais funções se estas estiverem salvas no mesmo arquivo.

Importando Módulos

- No VSCode pode ocorrer um erro de localização do arquivo.
- Para evitar isso, colocamos um código que força a executar o programa em uma determinada pasta. Assim facilita a importação dos módulos.

Importando Módulos

- import sys
- sys.path.append(r'c:\endereço_da_pasta')

Exercício 12

- Refaça os Exercício 11 removendo a função do código principal e os importe.

Salvar Dados Em Um Arquivo ".txt"

Salvar Dados Em Um Arquivo ".txt"

- Para trabalhar com arquivos no Python é muito simples.
- A função open( ) irá criar, escrever, ler um arquivo ".txt"

Salvar Dados Em Um Arquivo ".txt"

- A função open( ) abre o arquivo e retorna para o programa os dados lidos.

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")

arquivo.write("texto")

arquivo.close()
```

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

Variável que receber a leitura do arquivo

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

Função

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

Endereço e nome do arquivo a ser aberto em formato de string

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

Tipo de leitura:

W -> Escrever, sobrescrever, criar arquivo.

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

Variável que receber a leitura do arquivo

Função

Endereço e nome do arquivo a ser aberto em formato de string

Tipo de leitura:

W -> Escrever, sobrescrever, criar arquivo.

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
arquivo.write("texto")
```

Instrução para salvar texto no arquivo

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")

arquivo.write("texto")
```

Texto ou variável do tipo string que será salva no texto

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")
```

```
arquivo.write("texto")
```

Instrução para salvar texto no arquivo

Texto ou variável do tipo string que será salva no texto

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")

arquivo.write("texto")

arquivo.close()
```

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "w")

arquivo.write("texto")
```

```
arquivo.close()
```

Depois de abrir o arquivo, tem que fechar para salvar as alterações

Abrir Arquivo ".txt" E Recuperar Dados Salvos

Abrir Arquivo ".txt" E Recuperar Dados Salvos

- Para abrir arquivos é muito semelhante ao gravar em arquivos.
- Muda o "w" por "r"
- Use o laço for para recuperar os dados.

Abrir Arquivo ".txt" E Recuperar Dados Salvos

```
arquivo = open("nome_arquivo.txt", "w")

for linha in arquivo:

    print(linha)

arquivo.close()
```

Adicionar Dados Em Arquivos ".txt" Sem Sobrescrever O Original.

Adicionar Dados Em Arquivos ".txt" Sem Sobrescrever

- Para adicionar mais informação em um arquivo existente, deve-se usar o "a" no lugar do "r"
- É importante que no final da string a ser salva tenha a quebra de linha "\n".

Salvar Dados Em Um Arquivo ".txt"

```
arquivo = open("nome_arquivo.txt", "a")

arquivo.write("texto\n")

arquivo.close()
```

Exemplo

Exercício 13