



Rafael Vieira Coelho

rafaelvc2@gmail.com

PARTE 1

3 - Funções

O que são funções afinal?

- Funções são blocos de código nomeados, concebidos para realizar uma tarefa específica.
- Se precisar executar essa tarefa várias vezes durante seu programa, não será necessário digitar todo o código para a mesma tarefa repetidamente.
- Basta chamar a função (pelo nome) dedicada ao tratamento dessa tarefa e a chamada dirá a Python para executar o código da função.

Definindo uma Função

```
def greet_user():  
    print("Olá!")
```

```
def main():  
    greet_user()
```

```
main()
```

Olá!

Passando Uma Informação para a Função

```
def greet_user(user):  
    print("Olá, " + user + "!" )
```

```
def main():  
    name = input("Qual o seu nome, caro usuário?" )  
    greet_user(name)  
    greet_user("Mary")
```

```
main()
```

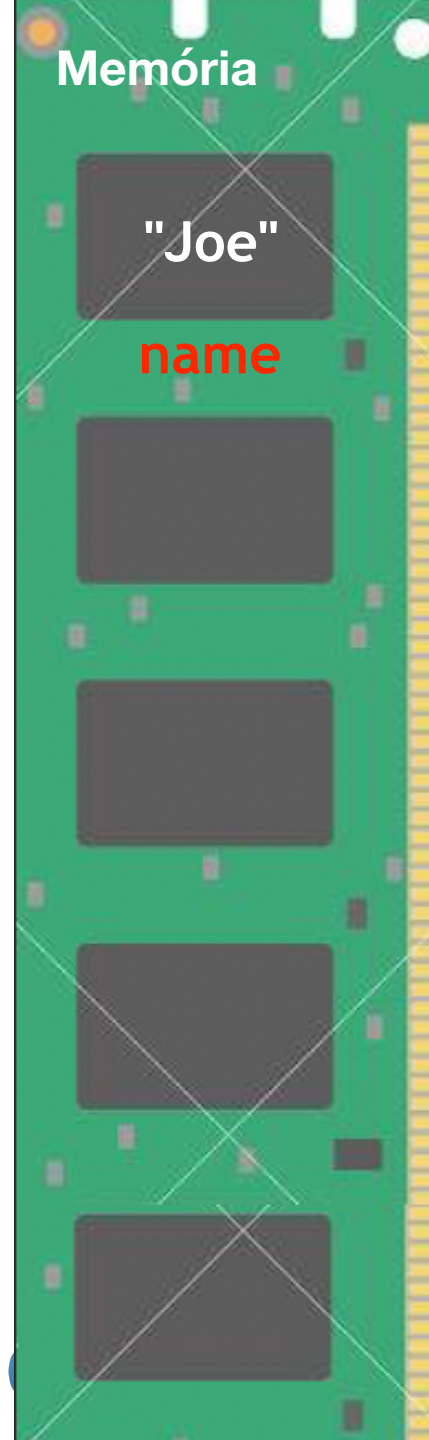
```
Qual o seu nome, caro usuário?  
Joe
```

```
Olá, Joe!  
Olá, Mary!
```

Memória

"Joe"

name



Passando Várias Informações para a Função

```
def greet_user(firstname, lastname):  
    print("Olá, " + firstname + " " + lastname + "!")
```

```
def main():  
    first = input("Qual o seu primeiro nome, caro usuário?")  
    last = input("E qual seria o seu sobrenome?")  
    greet_user(first, last)
```

main()

Qual o seu primeiro nome, caro usuário?

Joe

E qual seria o seu sobrenome?

Turner

Olá, Joe Turner!

Sempre verifique se a ordem dos argumentos em sua chamada de função corresponde à ordem dos parâmetros na definição da função.

"Joe"

first

"Turner"

last

Valores Padrão para os Parâmetros

```
def greet_user(user = 'Jack'):
    print("Olá, " + user + "!" )
```

```
def main():
    name = input("Qual o seu nome, caro usuário?")
    greet_user(name)
    greet_user()
```

```
main()
```

```
Qual o seu nome, caro usuário?
Joe
```

```
Olá, Joe!
Olá, Jack!
```

3.4 Criando novas funções

Até aqui, temos utilizado somente as funções que vêm com Python, mas também é possível adicionar novas funções.

No contexto de programação, função é uma sequência nomeada de instruções ou comandos, que realizam uma operação desejada.

A sintaxe para uma definição de função é:

```
def NOME_DA_FUNCAO( LISTA DE PARAMETROS ) :  
    COMANDOS
```

Observações sobre funções

1. Você pode chamar a mesma função repetidamente. Isso é muito comum, além de útil.
2. Você pode ter uma função chamando outra função.
3. Criar uma nova função pode tornar o programa menor, por eliminar código repetido.
4. Criar uma nova função permite que você coloque nome em um grupo de comandos. As funções podem simplificar um programa ao ocultar a execução de uma tarefa complexa por trás de um simples comando com nome significativo.

Exemplo

Exemplo3.py x

```
1  #!/usr/bin/env python
2  # coding: utf8
3
4  def novaLinha() :
5      print('')
6
7  def tresLinhas() :
8      novaLinha()
9      novaLinha()
10     novaLinha()
11
12     print ('Primeira Linha.')
13     tresLinhas()
14     print ('Segunda Linha.')
15
```

O que acontece quando executado o código ao lado?!

Primeira Linha

Segunda Linha

3.7 Variáveis e parâmetros são locais (escopo)

Quando você cria uma variável local dentro de uma função, ela só existe dentro da função e você não pode usá-la fora de lá. Por exemplo:

```
def concatDupla(parte1, parte2)
    concat = parte1 + parte2
    imprimeDobrado(concat)
```

Quando a função concatDupla termina, a variável concat é destruída!

Esta função recebe dois argumentos, concatena-os, e então imprime o resultado duas vezes. Podemos chamar a função com duas strings:

```
>>> canto1 = 'Eu sou feliz, '
>>> canto2 = 'e dai?! '
>>> concatDupla(canto1, canto2)
Eu sou feliz, e dai?! Eu sou feliz, e dai?!
```

3.9 Esqueleto de um programa Python

```
# função principal
def main():
    """
    Função principal, será a primeira a ser executado e
    será a responsável pela chamada de outras funções que
    por sua vez podem ou não chamar outras funções que
    por sua vez ...
    """
    # corpo da função main
    |
    | bloco de comandos
    |

# Declaração das funções
def f...
    """
    docstring da função f
    """
    # corpo da função f
    |
    | bloco de comandos
    |

def g...
    """
    docstring da função g
    """
    # corpo da função g
    |
    | bloco de comandos
    |

[...]

# início da execução do programa
main() # chamada da função main
```

5.1 Valores de retorno

Iremos escrever funções que retornam valores, as quais chamaremos de funções frutíferas, ou funções que dão frutos, na falta de um nome melhor.

O primeiro exemplo é `area`, que retorna a área de um círculo dado o seu raio:

```
import math
```

```
def area(raio):  
    temp = math.pi * raio**2  
    return temp
```

7.4 5.4 Funções booleanas

Funções podem retornar valores booleanos, o que muitas vezes é conveniente por ocultar testes complicados dentro de funções.

Por exemplo:

```
def ehDivisivel(x, y):  
    if x % y == 0:  
        return True # é verdadeiro (True), é divisível  
    else:  
        return False # é falso (False), não é divisível
```

Melhorando a função ehDivisivel()

Podemos tornar a função mais concisa se tirarmos vantagem do fato de a condição da instrução if ser ela mesma uma expressão booleana.

Podemos retorná-la diretamente, evitando totalmente o if:

```
def ehDivisivel(x, y):  
    return x % y == 0
```

```
def ehDivisivel(x, y):  
    if x % y == 0:  
        return True  
    else:  
        return False
```

Utilizando a função ehDivisivel()

Funções booleanas são frequentemente usadas em comandos condicionais:

```
if ehDivisivel(x, y):  
    print ("x é divisível por y")  
else:  
    print ("x não é divisível por y")
```

Como exercício, escreva uma função estaEntre(x, y, z) que retorne True se $x < y < z$ ou False, se não.

Importando Módulos (bibliotecas)

matematica.py

```
def soma(a, b):  
    return a + b  
  
def subtrai(a, b):  
    return a - b  
  
def multiplica(a, b):  
    return a * b  
  
def divide(a, b):  
    if b == 0:  
        return 0  
    return a / b
```

programa.py

```
import matematica  
  
def main():  
    v1 = int(input('Qual o primeiro valor?'))  
    v2 = int(input('Qual o segundo valor?'))  
    res = matematica.subtrai(v1, v2)  
    print('Resultado = ', res)  
  
main()
```


Dando um Apelido para um Módulo

matematica.py

```
def soma(a, b):  
    return a + b  
  
def subtrai(a, b):  
    return a - b  
  
def multiplica(a, b):  
    return a * b  
  
def divide(a, b):  
    if b == 0:  
        return 0  
    return a / b
```

programa1.py

```
import matematica as mat  
  
def main():  
    v1 = int(input('Qual o primeiro valor?'))  
    v2 = int(input('Qual o segundo valor?'))  
    res = mat.multiplica(v1, v2)  
    print('Resultado = ', res)  
  
main()
```

Importando Função de um Módulo

matematica.py

```
def soma(a, b):  
    return a + b  
  
def subtrai(a, b):  
    return a - b  
  
def multiplica(a, b):  
    return a * b  
  
def divide(a, b):  
    if b == 0:  
        return 0  
    return a / b
```

programa2.py

```
from matematica import soma  
  
def main():  
    v1 = int(input('Qual o primeiro valor?'))  
    v2 = int(input('Qual o segundo valor?'))  
    res = soma(v1, v2)  
    print('Resultado = ', res)  
  
main()
```

3.3 Funções matemáticas

Em matemática, você provavelmente já viu funções como seno (sen, sin em inglês) e logaritmo (log), e aprendeu a resolver expressões como $\sin(\pi/2)$ e $\log(1/x)$.

Antes de podermos usar as funções contidas em um módulo, temos de importá-lo:

```
>>> import math
```

Para chamar uma das funções, temos que especificar o nome do módulo e o nome da função, separados por um ponto.

Esse formato é chamado de notação de ponto:

```
>>> decibel = math.log10(17.0)
```

```
>>> angulo = 1.5
```

```
>>> altura = math.sin(angulo)
```

Solução de Problemas: Técnica Top-Down

1. Entender o problema a ser resolvido
2. Estabelecer o objetivo a ser alcançado
3. Dividir o problema (solução desconhecida) em problemas menores, com soluções mais simples (conhecidas) cujo total permita atingir o objetivo.
4. Continuar a subdividir os problemas gerados até que seja possível solucionar todos.
5. A solução do problema original é feita pela junção ordenada das soluções dos problemas finais.

Exemplo

Problema: Deseja-se construir um sistema automatizado para calcular as notas finais em todas as disciplinas de todos os alunos do Campus Farroupilha do IFRS.

1. Entender o problema a ser resolvido

O problema exige que se tenha as notas de cada aluno por disciplina. Com esses dados, deve-se calcular e exibir as notas finais.

2. Estabelecer o objetivo a ser alcançado

O objetivo é exibir as notas finais de todos os alunos em todas as disciplinas.

Exemplo

3. Dividir o problema (solução desconhecida) em problemas menores, com soluções mais simples (conhecidas) cujo total permita atingir o objetivo.

Início

Enquanto existirem disciplinas **Faça**

Calcular e exibir as notas finais de todas as disciplinas

Fim_enquanto

Fim

Exemplo

3. Dividir o problema (solução desconhecida) em problemas menores, com soluções mais simples (conhecidas) cujo total permita atingir o objetivo.

Início

Enquanto existirem disciplinas **Faça**

Enquanto existirem alunos na disciplina **Faça**

Calcular e exibir as notas finais deste aluno da disciplina atual

Fim_Enquanto

Fim_enquanto

Fim

Exemplo

4. Continuar a subdividir os problemas gerados até que seja possível solucionar todos.

Início

Enquanto existirem disciplinas **Faça**

Enquanto existirem alunos na disciplina **Faça**

Ler as notas de um aluno da disciplina atual

Calcular a nota final deste aluno

Escrever a nota final deste aluno

Fim_Enquanto

Fim_enquanto

Fim

Exemplo

5. A solução do problema original é feita pela junção ordenada das soluções dos problemas finais.

Início

Enquanto existirem disciplinas **Faça**

Enquanto existirem alunos na disciplina **Faça**

Inicializar somatório das notas

Enquanto existirem notas do aluno **Faça**

Ler a nota atual do aluno

Somar a nota atual com o somatório e atribuir a somatório

Fim_Enquanto

Calcular a média a partir do somatório

Escrever a nota final deste aluno

Fim_Enquanto

Fim_enquanto

Fim

Quais funções temos na solução?

Início

Enquanto existirem disciplinas **Faça**

Enquanto existirem alunos na disciplina **Faça**

Inicializar somatório das notas

Enquanto existirem notas do aluno **Faça**

Ler a nota atual do aluno

`ler_nota(mensagem)`

Somar a nota atual com o somatório e atribuir a somatório

Fim_Enquanto

`soma = somar_nota(nota, soma)`

Calcular a média a partir do somatório

Escrever a nota final deste aluno

`media = calcular_media(soma, total)`

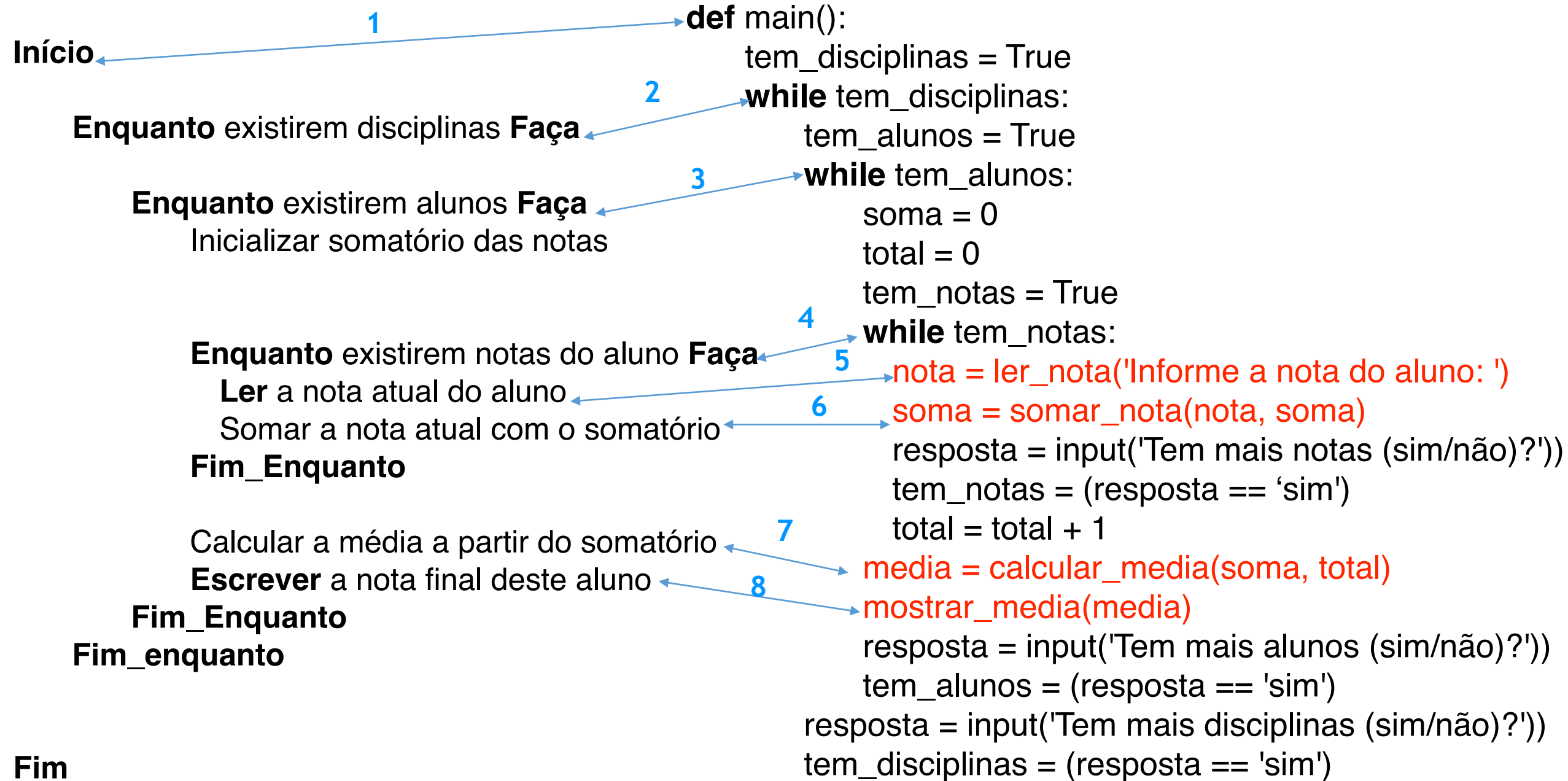
Fim_Enquanto

`mostrar_media(media)`

Fim_enquanto

Fim

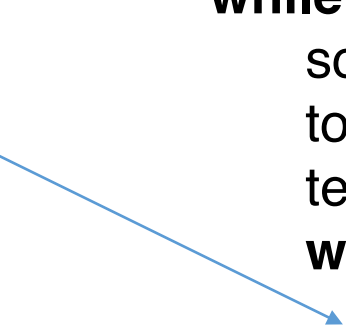
Quais funções temos na solução?



Função ler_nota()

```
def ler_nota(mensagem):  
    print(mensagem)  
    nota = float(input())  
    return nota
```

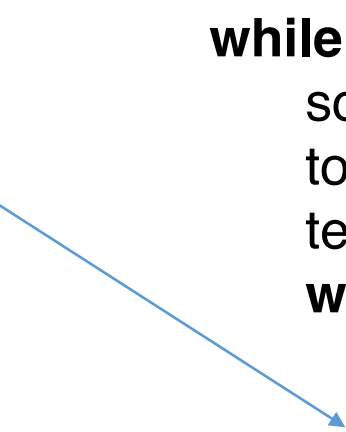
```
def main():  
    tem_disciplinas = True  
    while tem_disciplinas:  
        tem_alunos = True  
        while tem_alunos:  
            soma = 0  
            total = 0  
            tem_notas = True  
            while tem_notas:  
                nota = ler_nota('Informe a nota do aluno: ')  
                soma = somar_nota(nota, soma)  
                resposta = input('Tem mais notas (sim/não)?')  
                tem_notas = (resposta == 'sim')  
                total = total + 1  
            media = calcular_media(soma, total)  
            mostrar_media(media)  
            resposta = input('Tem mais alunos (sim/não)?')  
            tem_alunos = (resposta == 'sim')  
        resposta = input('Tem mais disciplinas (sim/não)?')  
        tem_disciplinas = (resposta == 'sim')
```



Função somar_nota()

```
def somar_nota(nota, soma):  
    total = soma + nota  
    return total
```

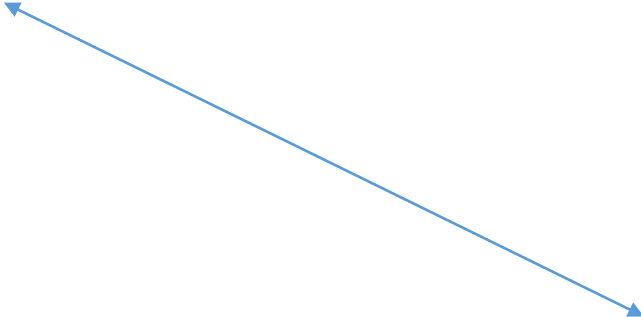
```
def main():  
    tem_disciplinas = True  
    while tem_disciplinas:  
        tem_alunos = True  
        while tem_alunos:  
            soma = 0  
            total = 0  
            tem_notas = True  
            while tem_notas:  
                nota = ler_nota('Informe a nota do aluno: ')  
                soma = somar_nota(nota, soma)  
                resposta = input('Tem mais notas (sim/não)?')  
                tem_notas = (resposta == 'sim')  
                total = total + 1  
            media = calcular_media(soma, total)  
            mostrar_media(media)  
            resposta = input('Tem mais alunos (sim/não)?')  
            tem_alunos = (resposta == 'sim')  
        resposta = input('Tem mais disciplinas (sim/não)?')  
        tem_disciplinas = (resposta == 'sim')
```



Função calcular_media()

```
def calcular_media(somatorio, num_notas):  
    resultado = somatorio / num_notas  
    return resultado
```

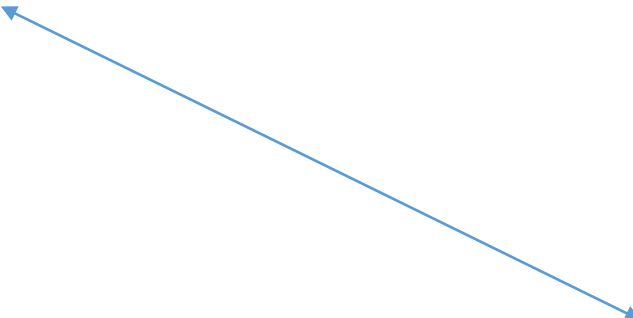
```
def main():  
    tem_disciplinas = True  
    while tem_disciplinas:  
        tem_alunos = True  
        while tem_alunos:  
            soma = 0  
            total = 0  
            tem_notas = True  
            while tem_notas:  
                nota = ler_nota('Informe a nota do aluno: ')  
                soma = somar_nota(nota, soma)  
                resposta = input('Tem mais notas (sim/não)?')  
                tem_notas = (resposta == 'sim')  
                total = total + 1  
            media = calcular_media(soma, total)  
            mostrar_media(media)  
            resposta = input('Tem mais alunos (sim/não)?')  
            tem_alunos = (resposta == 'sim')  
        resposta = input('Tem mais disciplinas (sim/não)?')  
        tem_disciplinas = (resposta == 'sim')
```



Função mostrar_media()

```
def mostrar_media(media_final):  
    print('Sua média final é ', media_final)
```

```
def main():  
    tem_disciplinas = True  
    while tem_disciplinas:  
        tem_alunos = True  
        while tem_alunos:  
            soma = 0  
            total = 0  
            tem_notas = True  
            while tem_notas:  
                nota = ler_nota('Informe a nota do aluno: ')  
                soma = somar_nota(nota, soma)  
                resposta = input('Tem mais notas (sim/não)?')  
                tem_notas = (resposta == 'sim')  
                total = total + 1  
                media = calcular_media(soma, total)  
                mostrar_media(media)  
                resposta = input('Tem mais alunos (sim/não)?')  
                tem_alunos = (resposta == 'sim')  
            resposta = input('Tem mais disciplinas (sim/não)?')  
            tem_disciplinas = (resposta == 'sim')
```



Exemplo Completo

```
def ler_nota(mensagem):
```

```
    print(mensagem)
```

```
    nota = float(input())
```

```
    return nota
```

```
def somar_nota(nota, soma):
```

```
    total = soma + nota
```

```
    return total
```

```
def calcular_media(somatorio, num_notas):
```

```
    resultado = somatorio / num_notas
```

```
    return resultado
```

```
def mostrar_media(media_final):
```

```
    print('Sua média final é ', media_final)
```

```
def main():
```

```
    tem_disciplinas = True
```

```
    while tem_disciplinas:
```

```
        tem_alunos = True
```

```
        while tem_alunos:
```

```
            soma = 0
```

```
            total = 0
```

```
            tem_notas = True
```

```
            while tem_notas:
```

```
                nota = ler_nota('Informe a nota do aluno: ')
```

```
                soma = somar_nota(nota, soma)
```

```
                resposta = input('Mais notas (sim/não)?')
```

```
                tem_notas = (resposta == 'sim')
```

```
                total = total + 1
```

```
            media = calcular_media(soma, total)
```

```
            mostrar_media(media)
```

```
            resposta = input('Tem mais alunos (sim/não)?')
```

```
            tem_alunos = (resposta == 'sim')
```

```
            resposta = input('Tem mais disciplinas (sim/não)?')
```

```
            tem_disciplinas = (resposta == 'sim')
```

```
main()
```


Exercícios

- 1) Faça um programa que leia 4 números e mostre a soma e a multiplicação dos mesmos.
- 2) Modularize o programa implementado no exercício 1. Ou seja, crie funções para cada uma das ações realizadas. Quando uma ação for repetida, ao invés de escrever o código novamente, deve-se chamar o nome da função criada que executa o código. Crie no mínimo 3 funções e o programa principal main.
- 3) Pesquise sobre funções da biblioteca math (matemáticas) e utilize duas delas em exemplos. <https://docs.python.org/3/library/math.html>

Exercícios

- 4) Faça uma função que determine o dobro de um número.
- 5) Faça uma função que determine o quádruplo de um número, usando a função do item anterior.
- 6) Faça uma função que, dado um total de segundos, calcule o total de horas, minutos e segundos.
- 7) Fornecidos três valores, a, b e c, implemente uma função que retorne quantos desses três são iguais. A resposta deve ser 2, se todos são iguais; 1, se dois são iguais ou 0, se todos são distintos entre si.