Subrotinas

- São estruturas que permitem agrupar comandos em blocos, que recebem um nome, e podem ser executados (ativados, chamados) em outras partes dos programas.

Parâmetros

- Semelhantes aos das funções matemáticas, são suportados pela subrotinas na maioria das linguagens de programação.
- Lembrar que em Python a passagem de parâmetros é sempre por cópia.
    - Mas se parâmetro for objeto, é passível de alteração...

Sintaxe

- São delimitados por parênteses – (e).
    - Mesmo que não haja parâmetros é necessário escrever os parênteses, ou seja, ( ).
- São separados por vírgulas (se houver mais de um).
- Não exigem a escrita do tipo de cada um deles.
- É possível especificar um valor padrão a ser usado em cada parâmetro caso não seja fornecido um valor na hora da ativação da subrotina.
    - Pode-se usar apenas um subconjunto deles na ativação.

Observações

- O local do retorno e a escolha do valor retornado são feitos com o comando **return**.
    - Se omitido, o retorno se dá ao final do seu código e o valor retornado será None.
    - Um comando return sem especificar um valor define o local do retorno e será o mesmo que return None.

```
import math
def Primo (num) : # Definição de função, num é o parâmetro…
    raiz = int (math.sqrt (num))
    i = 2
    while (i <= raiz) and ((num % i) != 0) :
        i = i + 1
        if i > raiz : i = 0
return i # Retorno do resultado da função…

numero = input ("Digite um número inteiro: ")
res = Primo (numero) # Chamada ou ativação da subrotina…
if res == 0 :
    print (numero, ‘é primo.')
else:
    print ('%d não é primo, %d é um divisor.' % (numero, res))
```

```
# Definição de procedimento usando valores padrão nos parâmetros
def repMens (m = ‘Hello!’, qtd = 1) :
    m = m * qtd # Valor de m só é alterado dentro da subrotina…
    print m
return # return é opcional: não retorna resultado de verdade...

repMens (‘Olá!’, 3) # Imprime ‘Olá!Olá!Olá!’
repMens (‘Olá!’) # Imprime ‘Olá!’
repMens ( ) # Imprime ‘Hello!’
repMens (qtd=3) # Imprime ‘Hello!Hello!Hello!’
res = repMens (‘Olá!’, 3) # Imprime ‘Olá!Olá!Olá!’ e res = None
repMens (4, 3) # Imprime ’12’!!!!
mens = ‘Mensagem’
repMens (mens, 3) # mens permanece com ‘Mensagem’...
```

1. Fazer um programa para:
   – Ler uma tabela com N profissões, onde
   • O valor de N é informado antes pelo usuário.

• Cada profissão é formada por um código (número positivo), um nome (String) e uma área (String).
• Leitura da tabela deve ser feita em subrotina.
– Depois o usuário fornecerá uma lista de códigos para que o programa informe o nome de cada profissão.
– Se o código da profissão não existir na tabela, mostrar a mensagem "Profissão Inexistente" e continuar.
– O programa pára com a digitação de um código inválido (negativo ou zero).

```
# Profissões - S1 - Tabela = Dicionário e função com resultado.
def preencheTab ( ) :
    n = input ('Digite o tamanho da tabela de profissões: ')
    while (not isinstance (n, int)) or (n < 1) :
        n = input ('Tamanho deve ser inteiro e positivo. Tente r
    tabela = { } # Cria a tabela na subrotina para retorná-la no
    for i in range (n) :
        codP = input ('Digite o código de uma profissão: ')
        # while que consiste codP foi omitido para subrotina cal
        nomeP = raw_input ('Digite o nome da profissão %d:\n' %
        areaP = raw_input ('Digite a área da profissão %d:\n' %
        tabela [codP] = (nomeP, areaP) # Inserção no dicionário
return tabela # O resultado retornado é o endereço do objeto tal

# Programa principal...
tab = preencheTab ( ) # variável tab receberá o endereço da tabe
print 'Tabela com %d profissões foi lida corretamente.' % (len(t
print 'Tabela ->', tab
codP = input ('Digite um código de profissão para busca (<=0 par
while codP > 0 :
    if codP in tab : # Verifica se a profissão existe na tabela
        nomeP, areaP = tab[codP] # Recupera os outros dados...
        print 'Profissão %d é %s e sua área é %s.' % (codP, nome
    else:
        print 'Profissão %d não existe na tabela.' % (codP)
```

```
    codP = input ('Digite outro código para busca (<=0 para para
print 'Fim de Programa'
```

```
# Profissões – S2 - Tabela = Dicionário e procedimento com parâm
def preencheTab (tabela) : # Recebe objeto tabela já criado para
    n = input ('Digite o tamanho da tabela de profissões: ')
    while (not isinstance (n, int)) or (n < 1) :
        n = input ('Tamanho deve ser inteiro e positivo. Tente n
    for i in range (n) :
        codP = input ('Digite o código de uma profissão: ')
        while (not isinstance (codP, int)) or (codP < 1) :
            codP = input ('Código deve ser inteiro e positivo. T
        nomeP = raw_input ('Digite o nome da profissão %d:\n' %
        areaP = raw_input ('Digite a área da profissão %d:\n' %
        tabela [codP] = (nomeP, areaP) # Inserção no dicionário
return # Este return pode ser omitido.

# Programa principal...
tab = { } # Cria a tabela e passa o endereço para a subrotina pr
preencheTab (tab)
print 'Tabela com %d profissões foi lida corretamente.' % (len(t
print 'Tabela ->', tab
codP = input ('Digite um código de profissão para busca (<=0 par
while codP > 0 :
    if codP in tab : # Verifica se a profissão existe na tabela
        nomeP, areaP = tab[codP] # Recupera os outros dados...
        print 'Profissão %d é %s e sua área é %s.' % (codP, nome
    else:
        print 'Profissão %d não existe na tabela.' % (codP)
    codP = input ('Digite outro código para busca (<=0 para para
print 'Fim de Programa'
```

Exercícios

1. Fazer um programa para:
   – Ler números inteiros positivos de até 5 dígitos (consistir) e imprimir quantas vezes o dígito 9 ocorre em cada um.
   – A leitura pára com número negativo ou zero.
   – Escrever subrotina que deve desmembrar o valor do número em seus 5 dígitos, retornando o resultado em uma lista de tamanho 5.
   – Escrever outra subrotina que usa a anterior e que:
   • Recebe como parâmetros um número positivo até 99999 e um algarismo inteiro (0 a 9), nesta ordem.
   • Retorne como resultado a quantidade vezes que o algarismo aparece no número.
2. Fazer um programa para achar todos os números palíndromos entre 100 e 5000.
   – Um número é palíndromo se ele tiver o mesmo valor quando escrito da direita para a esquerda. Ex: 17371.
   – Escreva e utilize uma subrotina cujo resultado é o valor recebido no parâmetro (int) escrito ao contrário.
   • Pode ser interessante utilizar a subrotina da questão anterior para desmembramento dos números.
3. Tente modificar programas feitos anteriormente para utilizar subrotinas.

```
# Fazer um programa para:
# – Ler números inteiros positivos de até 5 dígitos (consistir)
# – A leitura pára com número negativo ou zero.
# – Escrever subrotina que deve desmembrar o valor do número em

# – Escrever outra subrotina que usa a anterior e que:
# • Recebe como parâmetros um número positivo até 99999 e um alg
# • Retorne como resultado a quantidade vezes que o algarismo ap

print("\nQuestão 1")
def desmembrando(num):
    string = str(num)
    numeroDesmembrado = [0] * 5
    for i in range(len(string)):
```

```
        numeroDesmembrado[i] = string[i]
    return numeroDesmembrado

def contandoVezes(numero, algarismo=9):
    algarismoString = str(algarismo)
    contagem = 0
    for n in numero:
        if algarismoString == n:
            contagem += 1
    return contagem

numero = int(input("Digite um número inteiro positivo de até 5 d
while (numero <= 0) or (numero > 99999):
    numero = int(input("Número inválido. Digite um número intei
while numero > 0:
    numeroDesmembrado = desmembrando(numero)
    print(f"O número desmembrado em uma lista de tamanho 5 é {nu
    contagem = contandoVezes(numeroDesmembrado)
    print(f"O algarismo 9 aparece {contagem} vezes no número {nu

    print("\n::::::::::Nova leitura::::::::::")
    numero = int(input("Digite um número inteiro positivo de até
    while numero > 99999:
        numero = int(input("Digite um número inteiro positivo de

print("\n::::::::::Nova rotina::::::::::")
numero = int(input("Digite um número inteiro positivo de até 5 d
while (numero <= 0) or (numero > 99999):
    numero = int(input("Número inválido. Digite um número intei
algarismo = int(input("Digite um número entre 0 e 9: "))
while (algarismo < 0) or (algarismo > 9):
    algarismo = int(input("Número inválido. Digite um número ent
numeroDesmembrado = desmembrando(numero)
contagem = contandoVezes(numeroDesmembrado, algarismo)
print(f"O algarismo {algarismo} aparece {contagem} vezes no núme
```

```
# Fazer um programa para achar todos os números palíndromos entr
# – Um número é palíndromo se ele tiver o mesmo valor quando esc
# – Escreva e utilize uma subrotina cujo resultado é o valor rec
# • Pode ser interessante utilizar a subrotina da questão anteri
print("\nQuestão 2")

# – Escreva e utilize uma subrotina cujo resultado é o valor rec
def aoContrario(numero):
    numeroSTR = str(numero)
    numeroOposto = ''
    for i in range(len(numeroSTR)):
        numeroOposto = numeroSTR[i] + numeroOposto
    return numeroOposto

for i in range (100, 5001):
    numeroOposto = aoContrario(i)
    if numeroOposto == str(i):
        print(i)
```