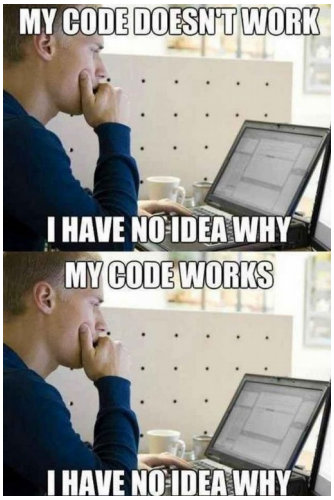
Lista de Exercícios – Funções
Prof: Yuri Frota

Funções

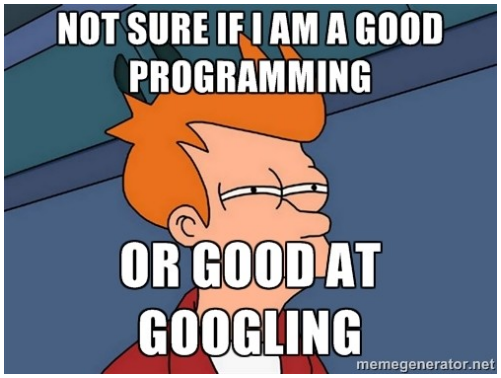
1)Faça um programa em Python para calcular as quatro operações, uma calculadora. Crie uma função que retorne a soma de dois números passados por parâmetro, outra para subtração, multiplicação e divisão respectivamente. Crie uma função chamada “interface_calculadora”, onde o programa deverá pedir dois números ao usuário e a operação desejada, se a a operação for soma deve ser chamado a função soma, e assim para as outras opções.




```
def soma(a,b):
   return a + b

def subtracao(a,b):
   return a - b

def multiplicacao(a,b):
   return a * b

def divisao(a,b):
   return a / b

def interface_calculadora():

   print "Menu:"
   print "1 Soma"
   print "2 Subtracao"
   print "3 Multiplicacao"
   print "4 Divisao"
   opc = input()

   inputA = input("Valor:")
   inputB = input("Valor:")

   if(opc == 1):
      print"O valor da soma: ", soma(inputA, inputB)

   if(opc == 2):
      print "O valor da subtracao: ", subtracao(inputA, inputB)

   if(opc == 3):
      print "O valor da multiplicao: ", multiplicacao(inputA, inputB)

   if(opc == 4):
      print "O valor da divisao: ", divisao(inputA, inputB)


interface_calculadora()
```

2)Escreva uma função em Python que receba uma lista de n números inteiros e retorne, para o usuário, o comprimento da maior sequencia crescente. Ex: na lista a = [6, 11, 4, 3, 5, 8, 10, 9, 6], o comprimento da maior sequencia crescente é 4 (pois 3,5,8 e 10 é a maior sequencia crescente). Já nesta lista b =[11, 9, 6, 4, 3], o comprimento da maior sequencia é 1.

```
def crescentSegmentLength(l):
    seq = 1
    aux = 0
    for i in range(1, len(l)):
        if l[i-1] < l[i]:
            seq = seq + 1
        else:
            seq = 1
        if seq > aux:
            aux = seq

    return aux
```



3) Faça um algoritmo que solicite ao usuário números e os armazene em um vetor de 20 posições. Crie uma função que recebe o vetor preenchido e substitua todas as ocorrências de valores negativos por zero, as ocorrências de valores menores do que 10 por 1 e as demais ocorrências por 2.

```
def altera(vet, tam):
    for i in range(tam):
        if vet[i] < 0:
            vet[i] = 0
        else:
          if vet[i] < 10:
            vet[i] = 1
          else:
            v[i] = 2
    return vet

tam = 39

v = [0]*tam
for i in range(tam):
    v[i] = input('Digite um valor: ')
altera(v,tam)
print v
```

4) Crie uma função que retorne o valor da expressão: 2/3 + 3/5 + 4/7 + 5/9 + ... + n/m, para um valor de n definido pelo usuário. No programa, verifique se o valor de n definido pelo usuário é positivo (antes de chamar a função) e, caso não seja, solicite outro valor até ser fornecido um valor positivo.

```
def sequ(n):
    aux1 = 2
    aux2 = 3.0
    soma = 0
```

```
    while aux1 <= num:
        print aux1, aux2
        soma = soma + aux1/aux2
        aux1 = aux1 + 1
        aux2 = aux2 + 2

    return soma

num = input('Digite um valor: ')
while num < 0:
    num = input('Digite um valor positivo: ')

res = sequ(num)
print res
```

5) Escreva uma função que recebe uma lista B com n elementos (sem repetições) e um índice k (onde 0 <= k < n) e tem como saída o índice do elemento mínimo entre B[k], B[k+1], ..., B[n-1]. Ex: B[6,2,9,4,6,11,2,3] e k=3 → índice 7

```
def mini(l, k):
    smallerIndex = k
    for i in range(k+1, len(l)):
        if(l[i] < l[smallerIndex]):
            smallerIndex = i
    return smallerIndex
```

Funções Recursivas

6) Faça uma função recursiva que receba um número inteiro positivo n e imprima a soma S definida como: S=n+(n-2)+(n-4)... (até n-x =< 0). Ex: n=6 → 12, n=10 → 30

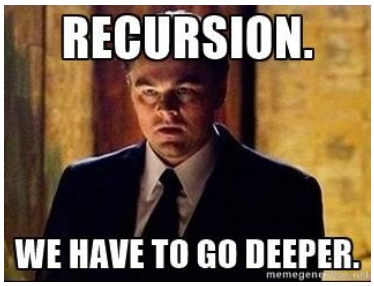

```
def sum_series(n):

 if n < 1:

    return 0

 else:

    return n + sum_series(n - 2)
```

7) Considere a seguinte fórmula para calcular o mdc (máximo divisor comum) de dois números inteiros positivos:

– mdc(a, b) = b, se b divide a (ou seja, a%b == 0)

– mdc(a, b) = mdc(b, a%b), caso contrário

Escreva uma função em Python que, dados dois números, retorne o máximo divisor comum entre eles.




```
def mdc(dividendo, divisor):
    if divisor == 0:
        return dividendo
    else:
        return mdc(divisor, dividendo % divisor)
```

8) Faça uma função recursiva que receba um número inteiro positivo n e imprima o n-ésimo número da sequencia de Fibonacci. Ex: n=5 → 5, n=2 → 1

```
def fibonacci(n):
  if n == 1 or n == 2:
    return 1
  else:
    return (fibonacci(n - 1) + (fibonacci(n - 2)))
```

9) Faça uma função recursica que receba um inteiro positivo n e imprima a soma dos dígitos de n. Ex: n=215 → 8, n=45 → 9

```
def sumDigits(n):
  if n == 0:
    return 0
  else:
    return n % 10 + sumDigits(int(n / 10))
```

10) Faça uma função recursica que receba um inteiro positivo n e imprima a soma harmônica de n definida como: 1/2+ 1/3 + 1/4 + ... + 1/n

```
def harmonic_sum(n):
  if n < 2:
    return 1
  else:
    return 1 / n + (harmonic_sum(n - 1))
```