Universidade Federal do Espírito Santo

Departamento de Computação e Eletrônica - CEUNES

Programação Funcional

Prof. Oberlan Romão

Lista de exercícios 3

1. Faça uma função lambda (anônima) que receba:

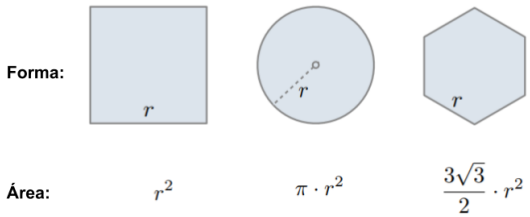
   a. Dois números a e b e retorne o valor de a^b;

   b. Dois números e retorne o resto da divisão do primeiro pelo segundo;

   c. Um número n e retorne `True` se n for par e `False`, caso contrário;

   d. Dois números e retorne `True` se o primeiro for divisível pelo segundo e `False`, caso contrário;

2. Defina uma função para calcular a área de um quadrado (`areaQuadrado`), de um círculo (`areaCirculo`) e de um hexágono (`areaHexagono`). Essa funções recebem a medida do lado/raio e retornam a área correspondente da figura ao qual estão associadas. Em seguida, defina uma função, chamada `imprime`, que recebe dois parâmetros: um valor correspondente ao tamanho do lado/raio e uma função. A função `imprime` deve, então, imprimir a área da figura usando a função que foi passada como parâmetro. Apresente diferentes exemplos de chamadas da função `imprime`.



   A função `imprime` deve, obrigatoriamente, ser uma função de alta ordem (*Higher Order Function*), ou seja, uma função que recebe como parâmetro outra(s) função(ões).

3. Faça uma função recursiva que:

   a. Imprima os números entre 1 e 10;

   b. Imprima todos os números palíndromos de 4 dígitos;

   c. Receba um número n e imprima os números de 1 a n;

   d. Receba dois número a e b e imprima os números naturais entre a e b;

   e. Receba dois inteiros a e b e retorna o valor a^b, sem usar o operador $**$, ou seja, $a * *b$ ou a função `pow(a,b)`;

   f. Receba um número inteiro n e retorna a quantidade de divisores de n;

   g. Receba um número inteiro n e retorna `True` se n for primo e `False`, caso contrário;

   h. Receba um número inteiro n e retorna a soma dos dígitos de n. Por exemplo, se $n = 12345$, sua função deve retornar 15.

4. A *sequência de Fibonacci* é uma sequência de termos que tem como os 2 primeiros termos, respectivamente, os números 1 e 1 e os número subsequente é a soma dos dois anteriores. A série de Fibonacci pode ser vista a seguir: $1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ 21,\ 34,\ 55,\ 89,\ 144,\ \ldots$

   Em termos matemáticos, a sequência é definida recursivamente pela fórmula abaixo,

$$F_n = \begin{cases} 1, & \text{se } n = 1 \text{ ou } n = 2 \\ F_{n-1} + F_{n-2}, & \text{se } n > 2 \end{cases}$$

   Faça uma função que imprima os n primeiros números da série de Fibonacci. Sua função deve, primeiramente, verificar se n é um número natural. Caso não seja, sua função deve exibir uma mensagem de erro e retornar `None`

5. Um *número perfeito* é aquele cuja soma de seus divisores, exceto ele próprio, é igual ao número. Por exemplo, 6 é perfeito porque 1 + 2 + 3 = 6. Escreva uma função, `ehPerfeito`, que receba um valor inteiro e retorne `True` se o número for perfeito e `False`, caso contrário. Em seguida, faça outra função, `perfeitos`, que imprima todos os números perfeitos de 1 a n. A função `perfeitos` deve utilizar a função `ehPerfeito`. Em ambas as funções deve-se verificar se o valor passado como argumento é um número natural. Faça uma função `main` que solicite ao usuário um número inteiro e que chama a função `perfeitos`.

6. Faça o rastreio (teste de mesa) dos programas abaixo. O que cada um imprime?

   a.
```
def funcao1(arg):
  if arg < 10:
    funcao1(arg + 1)
  else:
    print(arg)

funcao1(0)
```

   b.
```
def funcao2(arg):
  print(arg)
  if arg < 10:
    funcao2(arg + 1)

funcao2(0)
```

   c.
```
def funcao3(arg):
  if arg < 10:
    funcao3(arg + 1)
  print(arg)

funcao3(0)
```

7. O número de Euler e pode ser definido por: $e = \sum_{i=0}^{\infty} \frac{1}{n!}$.

   Faça uma função recursiva (chamada `euler`) para calcular o seu valor aproximado através dessa série. A função `euler` deve receber o número de termos a serem usados no cálculo, fazer o cálculo e retornar o resultado. Por exemplo, para 5 termos, o resultado aproximado é o valor de

$$e = \sum_{i=0}^{4} \frac{1}{n!} = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} = 2,70833$$

Em seguida, faça uma função `main` que leia o número de termos a serem usados no cálculo e escreva o resultado. Veja um exemplo:

```
Numero de termos: 5
Resultado: 2,70833
```