TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica **Área Académica de Ingeniería en Computadores** **Programa de Licenciatura en Ingeniería en Computadores** **Curso:** CE-1101 Introducción a la programación **Profesor:** Lic. Ing. Fabián Zamora Ramírez **Semestre:** I, 2018	**Práctica de temas:** • Programación recursiva con Python. Números. (1)

INDICACIONES:

Para todos los siguientes ejercicios muestre su solución en lenguaje de programación Python. Toda función debe estar documentada con una descripción breve de lo que realiza y mostrar sus entradas (E), salidas (S) y restricciones (R). Si no se especifican restricciones, debe realizar las restricciones lógicas según el problema. Recuerde utilizar nombres significativos para sus funciones y variables. En caso de no cumplir con lo anterior se calificará que nota de cero.

1. La operación de multiplicar dos números a * b, cuando b es un número entero, puede calcularse recursivamente por medio de b - 1 sumas sucesivas del número a. Por ejemplo, la multiplicación 8 * 4, podría verse como: 8 + 8 + 8 + 8 (tres sumas sucesivas de 8).

   Escriba una función recursiva **multi_sumas(num1, num2)** que reciba dos números e implemente la multiplicación por medio de sumas sucesivas. La ejecución de la función debe mostrar resultados como los siguientes:

   a. >>> multi_sumas(8, 4) -> 32
   b. >>> multi_sumas(-9, 5) -> -45
   c. >>> multi_sumas(-9, -5) -> 45
   d. >>> multi_sumas(9, -5) -> -45

2. Escriba una función recursiva **suma_dig(num)** que reciba un número que puede ser entero o real y sume sus dígitos. La función debe brindar resultados como los que se presentan en los siguientes ejemplos:

   a. >>> suma_dig (4123) -> 10
   b. >>> suma_dig (10001) -> 2

   c. >>> suma_dig (122.45) -> 14

3. Escriba una función recursiva **suma_pares(n)** que encuentre la suma de los enteros positivos pares desde 0 hasta N.
4. Escriba una función recursiva **suma_impares(n)** que encuentre la suma de los enteros positivos impares desde 0 hasta N.
5. Escriba una función recursiva **invierte(num)** que reciba un numero entero y devuelva el numero con sus dígitos invertidos.
   a. >>> invierte(123) -> 321
   b. >>> invierte(-123) -> -321
6. Escriba una función recursiva **mcd(a,b)** que calcule el máximo común divisor de dos números.
   a. >>>mcd(5, 25) -> 5
   b. >>>mcd(100, 525) -> 25
7. Escriba una función recursiva **múltiplos(n)** que muestre la tabla de multiplicar de n. N puede tomar valores del 1 al 10.
   a. >>>múltiplos(9)
      i. 9 x 1 = 9
      ii. 9 x 2 = 18
      iii. 9 x 3 = 27
      iv. 9 x 4 = 36
      v. 9 x 5 = 45
      vi. 9 x 6 = 54
      vii. 9 x 7 = 63
      viii. 9 x 8 = 72
      ix. 9 x 9 = 81
      x. 9 x 10 = 90
8. Escriba una función recursiva **potencia_dos(n)** que muestre todas las potencias de 2, desde la 0-ésima hasta la n-ésima.
   a. >>>potencia_dos(10)
      i. >>>1 2 4 8 16 32 64 128 256 512 1024

9. Escriba una función recursiva **es_binario(n)** que reciba un numero entero y retorne True si el numero contiene solo dígitos del sistema binario.
10. Escriba una función recursiva **es_base(n, base)** que reciba un numero entero y una base y retorne True si el numero contiene solo dígitos de esa base. La base puede ser: 2, 4, 8, 10 o 16.
11. ejeEscriba una función recursiva llamada **bin_to_dec(num)** que reciba un número entero en binario y devuelva su equivalente en base decimal.
    a. >>> bin_to_dec(111) -> 7
    b. >>> bin_to_dec(100000) -> 32
12. Escriba una función llamada recursiva **dec_to_bin(num)** que reciba un número entero en base decimal y devuelva su equivalente en binario.
    a. >>> dec_to_bin(7) -> 111
    b. >>> dec_to_bin (32) -> 100000
13. Escriba una función recursiva **str_contains(cadena, caracter)** que reciba una cadena de caracteres y retorne True si en la cadena se encuentra al menos 3 veces el carácter.
    a. >>> str_contains(“abcdefabcda”,”a”) -> True
    b. >>> str_contains(“12342”,”2”) -> False
14. Escriba una función recursiva **str_in_list(cadena, lista)** que reciba una cadena de caracteres y retorne True si la cadena solo contiene caracteres que se encuentran en la lista.
    a. >>> str_in_list(“abcdddabc”,[“a”, “b”, “c”, “d”]) -> True
    b. >>> str_in_list(“101100”,[“1”, “0”]) -> True
    c. >>> str_in_list(“1011030”,[“1”, “0”]) -> False
15. Escriba una función recursiva **calificaciones(lista)** que reciba una lista de calificaciones y retorne una tupla con la nota más alta, la más baja y el promedio.
    a. >>> calificaciones([1,2,3,4,5,6]) -> (6, 1, 3.5)