**[Ejercicios en python (Parte 1)](http://www.pythondiario.com/2013/05/ejercicios-en-python-parte-1.html" \o "Permanent Link to Ejercicios en python (Parte 1))**

|  |
| --- |
|  |

Definir una función max() que tome como argumento dos números y devuelva el mayor de ellos. (Es cierto que python tiene una función max() incorporada, pero hacerla nosotros mismos es un muy buen ejercicio.

|  |  |
| --- | --- |
| #! /usr/bin/env python # -\*- coding: utf-8 -\*-  def max (n1, n2): if n1 < n2: print n2 elif n2 < n1: print n1 else: print "Son iguales" | def maximo(x,y):   return x if x > y else y |

2- Definir una función max\_de\_tres(), que tome tres números como argumentos y devuelva el mayor de ellos.

|  |  |
| --- | --- |
| #! /usr/bin/env python # -\*- coding: utf-8 -\*- def max\_de\_tres (n1, n2, n3): if n1 > n2 and n1 > n3: print n1 elif n2 > n1 and n2 > n3: print n2 elif n3 > n1 and n3 > n2: print n3 else: print "Son iguales" | def max\_de\_tres\_1(x,y,z): m = maximo(x,y) return maximo(m,z)  # Y sin llamar a la funcion del punto1 def max\_de\_tres\_2(x,y,z): if x > y and x > z: return x elif y > x and y > z: return y return z |

3- Definir una función que calcule la longitud de una lista o una cadena dada. (Es cierto que python tiene la función len() incorporada, pero escribirla por nosotros mismos resulta un muy buen ejercicio.

|  |  |
| --- | --- |
| #! /usr/bin/env python # -\*- coding: utf-8 -\*-  def largo\_cadena (lista): cont = 0 for i in lista: cont += 1 return cont | def longlista(lista):  print(len(lista)) |

4- Escribir una función que tome un carácter y devuelva True si es una vocal, de lo contrario devuelve False.

|  |  |
| --- | --- |
| #! /usr/bin/env python # -\*- coding: utf-8 -\*-  def es\_vocal (x): if x == "a" or x == "e" or x == "i" or x == "o" or x == "u": return True elif x == "A" or x == "E" or x == "I" or x == "O" or x == "U": return True else: return False | def esVocal(caracter):  vocales = ['a', 'e','i','o','u','A', 'E', 'I', 'O', 'U']  return caracter in vocales |

5- Escribir una funcion sum() y una función multip() que sumen y multipliquen respectivamente todos los números de una lista. Por ejemplo: sum([1,2,3,4]) debería devolver 10 y multip([1,2,3,4]) debería devolver 24.  
  
QUIERO SUMARLE A TODOS LOS NUMEROS DE LA LISTA UN NUMERO

|  |  |
| --- | --- |
| **Ejercicio 5**  #! /usr/bin/env python # -\*- coding: utf-8 -\*-  def sum (lista): suma = 0 for i in lista: suma += i return suma  def multip (lista): multiplicacion = 1 for i in lista: multiplicacion \*= i return multiplicacion |  |

6- Definir una función inversa() que calcule la inversión de una cadena. Por ejemplo la cadena "estoy probando" debería devolver la cadena "odnaborp yotse"

|  |  |
| --- | --- |
| #! /usr/bin/env python  # -\*- coding: utf-8 -\*-  #range(start, stop[, step]) -> range object  def inversa1(cadena):  invertida = ""  start=len(cadena)-1  for i in range(start,-1,-1):  invertida += cadena[i]  return invertida | def inversa(palabra):  return palabra[::-1] |

7 - Definir una función es\_palindromo() que reconoce palíndromos (es decir, palabras que tienen el mismo aspecto escritas invertidas), ejemplo: es\_palindromo ("radar") tendría que devolver True.

|  |  |
| --- | --- |
| def palindro(cadena):  lon=len(cadena)-1  cont=0  bcadena=(cadena[cont] == cadena[lon])  while (bcadena):  lon-=1  cont+=1  bcadena=(cadena[cont] == cadena[lon])  break  return bcadena | def es\_palindromo(word): return str(word) == str(word)[::-1]  No es necesario str  def es\_palindromo(palabra):  return palabra==palabra[::-1] |

8- Definir una función superposicion() que tome dos listas y devuelva True si tienen al menos 1 miembro en común o devuelva False de lo contrario. Escribir la función usando el bucle for anidado.

NO ENTIENDO POR QUÉ FUNCIONA

|  |  |
| --- | --- |
| #! /usr/bin/env python # -\*- coding: utf-8 -\*-  def superposicion (lista1, lista2): for i in lista1:  for x in lista2:  if i == x:  return True  return False  #>>> superposicion([1,2],[1,2,3])  True  >>> superposicion([1,0,3],[1,2,3])  True |  |

9- Definir una función generar\_n\_caracteres() que tome un entero n y devuelva el caracter multiplicado por n. Por ejemplo: generar\_n\_caracteres(5, "x") debería devolver "xxxxx".

|  |  |
| --- | --- |
| def generar\_n\_caracteres(n,car):  print (n \* car) |  |

10- Definir un histograma procedimiento() que tome una lista de números enteros e imprima un histograma en la pantalla. Ejemplo: procedimiento

([4, 9, 7]) debería imprimir lo siguiente:

\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| #! /usr/bin/env python  # -\*- coding: utf-8 -\*-  def generar\_n\_caracteres(n,car):  print (n \* car)  def histogramallama (lista):  for i in lista:  generar\_n\_caracteres(i,"\*") | #! /usr/bin/env python  # -\*- coding: utf-8 -\*-  def Histograma (lista):  for i in lista:  print (i\*”\*”) |

Los números de empleados en una empresa están codificados de 100 a 199. Asuma que en un vector se ha almacenado los salarios/hora y en otro vector las horas trabajadas de los empleados de dicha empresa. Elabore un programa que lea n números de empleados y que calcule el salario bruto de cada empleado leído.

Es leer un archivo la cual contara todo las palabras y si la palabra esta repetida contar cuantas veces se repite y ponerlo en un diccionario, para ya después en listar en un archivo nuevo. con la palabra y el numero de repitiente