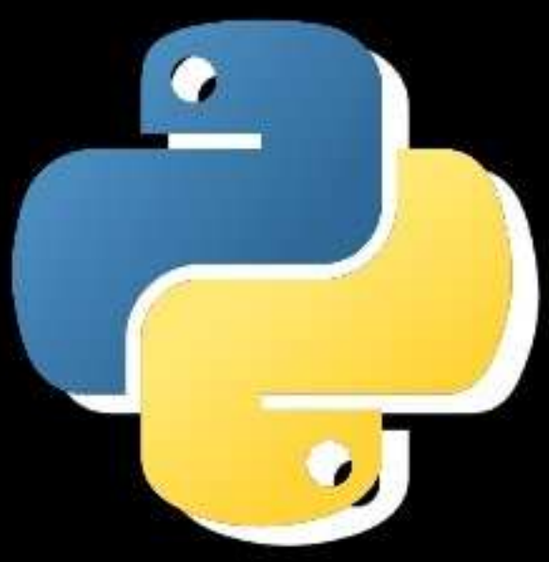

python™

Inteligencia Artificial

Lenguaje Phyton.

Universidad Nacional de Ingeniería
FACULTAD DE CIENCIAS Y SISTEMAS – DEPARTAMENTO DE INFORMÁTICA

Introducción.

Programar es el arte de hacer que cualquier dispositivo inteligente ejecute las instrucciones que se les suministra en un idioma que el dispositivo pueda entender y que el dispositivo interpreta literalmente.

Cada lenguaje de programación posee una forma propia que le permite al programador darle instrucciones básicas a ese dispositivo inteligente.

Python fue diseñado a finales de la década de los ochenta por Guido van Rossum. Un lenguaje de programación de muy alto nivel, con una sintaxis muy clara y una apuesta firme por la legibilidad del código. Sin duda muy versátil, fuertemente tipado, imperativo y orientado a objetos, aunque contiene también características que lo convierten en un lenguaje de paradigma funcional.

Python posee una simplicidad intuitiva tal, que con unas pocas líneas de instrucción podemos ejecutar actividades complejas que en otro lenguaje requerirían muchas más líneas de código o mayor número de instrucciones.

Lenguajes de programación con tal facilidad se denominan de alto nivel.

Para tener una idea de la creciente comunidad que usa Python, puede visitar y explorar https://www.python.org/community.

Practicando a domar a Python.

1. Imprimir “Hola mundo” por pantalla.

```
#!/usr/bin/python3
print("Hola mundo")
```

2. Crear dos variables numéricas, sumarlas y mostrar el resultado.

```
#!/usr/bin/python3
variable1 = 5
variable2 = 6
suma = variable1 + variable2
print("La suma de ",variable1,"+",variable2,"=",suma)
```

3. Mostrar el precio del IVA de un producto con un valor de C$100.00 y su precio final.

```
#!/usr/bin/python3
IVA = 0.15
precioProducto = 100
precioIVA = precioProducto * IVA
print("El precio del IVA es", precioIVA, "C$")
print("El precio final es", (precioIVA+precioProducto) ,"C$")
```

4. De dos números, saber cuál es el mayor.

```
#!/usr/bin/python3
a = 7
b = 5
if( a<b ):
    print("A es menor que B")
else:
    print("B es menor que A")
```

5. Crea una variable numérica y si esta entre 0 y 10, mostrar un mensaje indicándolo.

```
#!/usr/bin/python3
a = 5
if(a>=1 and a<=10):
    print("Está entre 1 y 10")
else:
```

```
    print("No está en ese rango")
```

6. Añadir al ejercicio anterior, que si está entre 11 y 20 muestre otro mensaje diferente y si está entre 21 y 30 otro mensaje.

```
#!/usr/bin/python
a = 35
if(a>=1 and a<=10):
    print("Está entre 1 y 10")
elif(a>=11 and a<=20):
    print("Está entre 11 y 20")
elif(a>=21 and a<=30):
    print("Está entre 21 y 30")
else:
    print("No está en ese rango")
```

7. Usar un while para mostrar los números del 1 al 100.

```
#!/usr/bin/python3
i = 1
while( i<=100 ):
    print(i)
    i+=1
print("Fin del bucle")
```

8. Mostrar con un for los números del 1 al 100.

```
#!/usr/bin/python3
for i in range(1,101):
    print(i)
```

9. Mostrar los caracteres de la cadena “Hola mundo”.

```
#!/usr/bin/python3
for i in "Hola mundo":
    print(i)
```

10. Mostrar los números pares entre 1 al 100.

```
#!/usr/bin/python3

#1º forma
```

```
print("1 forma")
for i in range(1,101):
    if( (i%2)==0 ):
        print(i)
print("")

#2º forma
print("2 forma")
for i in range(2,101,2):
    print(i)
```

11. Generar un rango entre 0 y 10.

```
#!/usr/bin/python3
rango = list( range(10) )
print(rango)
```

12. Generar un número entre 5 y 10.

```
#!/usr/bin/python3
rango = list(range(5,10))
print(rango)
```

13. Generar un rango de 10 a 0.

```
#!/usr/bin/python3
rango = list(range(10,0,-1))
print(rango)
```

14. Generar un rango de 0 a 10 y de 15 a 20, incluidos el 10 y 20.

```
#!/usr/bin/python3
rango1 = list(range(0,11))
rango2 = list(range(15,21))
final = rango1 + rango2
print(final)
```

15. Generar un rango desde 0 hasta la longitud de la cadena “Hola mundo”.

```
#!/usr/bin/python3
rango = list( range(0, len("Hola mundo")))
print(rango)
```

16. Pide dos cadenas por teclado, muestra ambas cadenas con un espacio entre ellas y con los 2 primeros caracteres intercambiados. Por ejemplo, hola mundo pasaría a mula hondo.

```
#!/usr/bin/python3
cadena1 = input("Dame la primera cadena: ")
cadena2 = input("Dame la segunda cadena: ")
print( cadena2[:2] + cadena1[2:] + " " + cadena1[:2] + cadena2[2:] )
```

17. Pide una cadena e indica si es un palíndromo o no.

```
#!/usr/bin/python3
cadena1 = input("Dame una cadena: ")
cadena_al_reves = cadena1[::-1]
print(cadena_al_reves)
if( cadena1 == cadena_al_reves ):
    print("Es palíndromo")
else:
    print("No es palíndromo")
```

18. Adivina el número entre 1 y 100.

```
#!/usr/bin/python3
from random import *
def generaNumeroAleatorio(minimo,maximo):
    try:
        if minimo > maximo:
            aux = minimo
            minimo = maximo
            maximo = aux
        return randint(minimo, maximo)
    except TypeError:
        print("Debes escribir números")
        return -1
numero_buscado = generaNumeroAleatorio(1,100)
encontrado = False
intentos = 0
while not encontrado:

    numero_usuario = int(input("Introduce el número buscado: "))
     if numero_usuario > numero_buscado:
```

```
        print("El número que buscas es menor")
        intentos = intentos +1
    elif numero_usuario < numero_buscado:
        print("El número que buscas es mayor")
        intentos = intentos +1
    else:
        encontrado = True
        print("Has acertado el número correcto es " , numero_usuario, " te ha llevado ",
intentos," intentos ganar en este juego.")
```

19. Definir una función max() que tome como argumento dos números y devuelva el mayor de ellos.

```
#! /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"
```

20. 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"
```

21. Definir una función que calcule la longitud de una lista o una cadena dada.

```
def largo_cadena (lista):
```

```
    cont = 0
    for i in lista:
        cont += 1
    return cont
```

22. 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
```

23. Escribir una función 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.

```
#! /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
```

24. 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 -*-
def inversa (cadena):
```

```
invertida = ""
cont = len(cadena)
indice = -1
while cont >= 1:
    invertida += cadena[indice]
    indice = indice + (-1)
    cont -= 1
return invertida
```

25. 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.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def inversa (cadena):
    invertida = ""
    cont = len(cadena)
    indice = -1
    while cont >= 1:
        invertida += cadena[indice]
        indice = indice + (-1)
        cont -= 1
    return invertida

 def es_palindromo (cadena):
    palabra_invertida = inversa (cadena)
    indice = 0
    cont = 0
for i in range (len(cadena)):
    if palabra_invertida[indice] == cadena[indice]:
        indice += 1
        cont += 1
    else:
        print "No es palindromo"
        break

if cont == len(cadena): #Si el contador = a la cantidad de letras de la cadena
    print "Es palindromo" # es porque recorrió todo el ciclo for y todas las
                                    # letras son iguales
```

26. 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.

```
#! /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
```

27. 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".

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def generar_n_caracteres (n, caracter):
    print n * caracter
```

28. 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:

```
****
*********
*******
```

```
#La solución sería:
#! /usr/bin/env python
# -*- coding: utf-8 -*-

def procedimiento (lista):
    for i in lista:
        print i * "x"
```

29. La función max() y la función max_de_tres(), solo van a funcionar para 2 o 3 números. Supongamos que tenemos más de 3 números o no sabemos cuántos

números son. Escribir una función max_in_list() que tome una lista de números y devuelva el más grande.

30. Escribir una función mas_larga() que tome una lista de palabras y devuelva la más larga.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def max_in_list(lista):
    inicio = 0
    for i in lista:
    if i > inicio:
        inicio = i
    return inicio
```

31. Escribir una función filtrar_palabras() que tome una lista de palabras y un entero n, y devuelva las palabras que tengan más de n caracteres.

```
#! / Usr / bin / env python
# - * - Codificación: utf-8 - *
 def filtrar_palabras(lista, n):
    for i in lista:
        if len(i) > n:
            print i
```

32. Escribir un programa que diga al usuario que ingrese una cadena. El programa tiene que evaluar la cadena y decir cuántas letras mayúsculas tiene.

```
#! / Usr / bin / env python
# - * - Codificación: utf-8 - *

def c_mayusculas (cadena):
    cont = 0
    for i in cadena:
        if i != i.lower(): #Recordar que lower() convierte una cadena en minúsculas
            cont += 1
    print "La cadena tiene", cont, "mayuscula/s"
```

33. Construir un pequeño programa que convierta números binarios en enteros.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
```

```
def aDecimal(numeroBin):
    numeroBin = str(numeroBin)
    decimal = 0
    exp = len (numeroBin) -1
    for i in numeroBin:
        decimal += (int(i) * 2**(exp))
        exp = exp - 1
    return decimal
```

34. Escribir un pequeño programa donde:
    - Se ingresa el año en curso.
    - Se ingresa el nombre y el año de nacimiento de tres personas.
    - Se calcula cuántos años cumplirán durante el año en curso.
    - Se imprime en pantalla.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def main():
    a_curso = input ("Ingresa el año en curso: ")
    for i  in range (3):
        nombre = raw_input ("Nombre de la persona: ")
        nacimiento = input ("Año de nacimiento: ")
        print  nombre, "cumple", (a_curso - nacimiento), "años en el", a_curso
```

35. Definir una tupla con 10 edades de personas. Imprimir la cantidad de personas con edades superiores a 20.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def mayora20 (tup):
    cont = 0
    for i in tup:
        if i > 20:
            cont += 1
    print "Hay", cont, "numeros mayores a 20"
```

36. Definir una lista con un conjunto de nombres, imprimir la cantidad de comienzan con la letra a. También se puede elegir al usuario la letra a buscar.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
```

```
def main():
    x = input ("Cuantos nombres quieres ingresar?: ")
    lista = []
    for i in range(x):
        a = raw_input ("Ingresa el nombre: ")
        lista.append (a)

    print ""
    comienzo = raw_input ("Con que letra empieza el nombre?: ")
    cont = 0
    for i in lista:
        if i[0] == comienzo.lower() or i[0] == comienzo.upper() :
            cont += 1
    return cont
```

37. Crear una función contar_vocales(), que reciba una palabra y cuente cuantas letras "a" tiene, cuantas letras "e" tiene y así hasta completar todas las vocales. Se puede hacer que el usuario sea quien elija la palabra.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def contar_vocales(cadena):
    cadena = cadena.lower()
    vocales = "aeiou"

    for x in vocales:
        contador = 0
        for i in cadena:
            if i == x:
                contador += 1
        print "Hay %d %s." % (contador, x)
```

38. Escriba una función es_bisiesto() que determine si un año determinado es un año bisiesto. Un año bisiesto es divisible por 4, pero no por 100. También es divisible por 400.

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
def es_bisiesto():
    print "Comprueba años bisiestos"
    a = input ("Escriba un años y le dire si es bisiesto: ")
    if a % 4 == 0 and (not(a % 100 == 0)):
```

```
    print "El año", a, "es un año bisiesto porque es multiplo de 4"
elif a % 400 == 0:
    print "El año", a, "es un año bisiesto porque es multiplo de 400"
else:
    print "El año", a, "no es bisiesto"
```


python
TM