

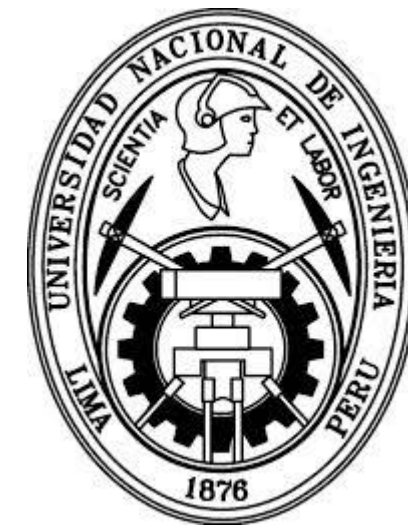


Métodos Numéricos

PhD. Alejandro Paredes

FC-UNI

Cronograma actividades



Inicio de clases (1ra semana)	28 de agosto
Matrícula de Traslados Internos, Externos, Graduados, Titulados, Convenios y Rezagados.	Del 28 de agosto al 01 de setiembre
Verificación de Matrícula	Del 28 de agosto al 01 de setiembre
Retiro parcial de cursos (5ta semana)	Del 25 al 29 de setiembre
Retiro Total	Hasta el 13 de octubre
Exámenes Parciales (8va semana)	Del 16 al 21 de octubre
EXPOCIENCIA 2023	13 al 17 de noviembre
Retiro total, excepcional (15ta semana)	Del 04 al 07 de diciembre
Examen Final (16ta semana)	Del 11 al 16 de diciembre
Examen Sustitutorio (18va semana)	Del 26 al 29 de diciembre

Organización del Curso



- Sistema de evaluación :
- PC 1 : Semana 4 => 21.09.2023
- PC 2 : Semana 7 => 12.09.2023
- PC 3 : Semana 11 => 09.11.2023
- PC 4 : Semana 15 => 07.12.2023
- Examen Parcial : 19.10.2023
- Examen Final : 14.12.2023
- Examen Sustitutorio : 28.12.2023
- Se elimina 1 PC
- **Nota final = (PP + EP + EF) /3**



Organización del Curso

Silabo semanalizado

- S01 : Introducción, Linux, scripts Python.
- S02 : Ecuaciones no lineales.
- S03 : Sistemas lineales – Métodos directos.
- S04 : Sistemas lineales – Métodos iterativos.
- S05 : Interpolación Polinomial.
- S06 : Ajuste de curvas.
- S07 : Integración y diferenciación numéricas.
- S08 : EDO
- S09 : Problemas condiciones de frontera.
- S10 : EDP
- S11 : EDP-parabólicas.
- S12 : EDP-hiperbólicas.
- S13: Elementos finitos 1.
- S14 : Elementos finitos 2.



Organización del Curso

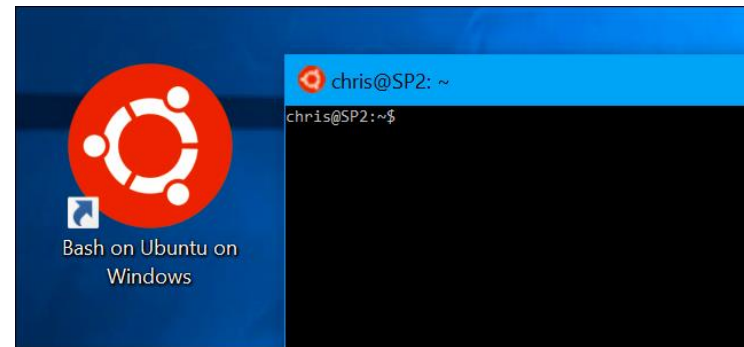
- Bibliografía
 - Chapra, Applied Numerical Methods for Engineers and Scientists.
 - Chapra, Applied Numerical Methods for Scientists.
 - Alejandro Garcia, Numerical Methods for Physics.
 - Mark Newman, Computational Physics.
 - Numerical Recipes Fortran 90.

Accesso al SO Linux

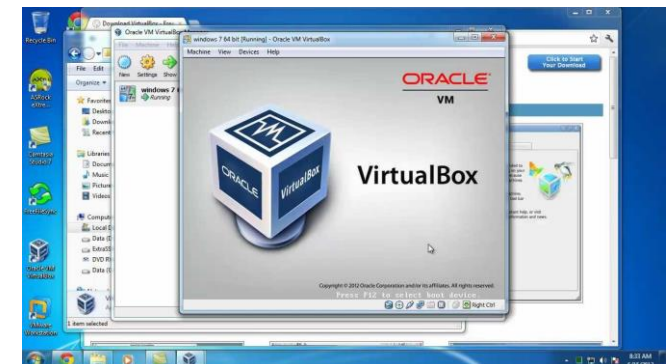
- Formatear la computadora e instalar Ubuntu.
Más recomendable !!!



- Si tienes Windows 10:
Windows Subsystem for Linux
Bash en Ubuntu en Windows

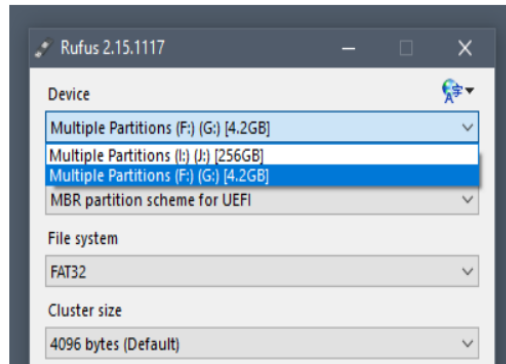
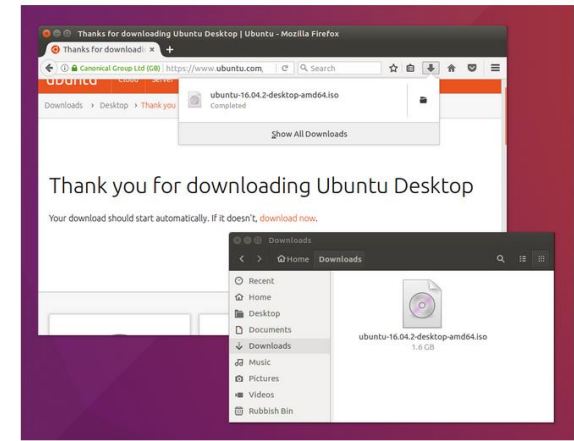


- Si tienes Windows < 10:
Virtual box (genera una maquina virtual)



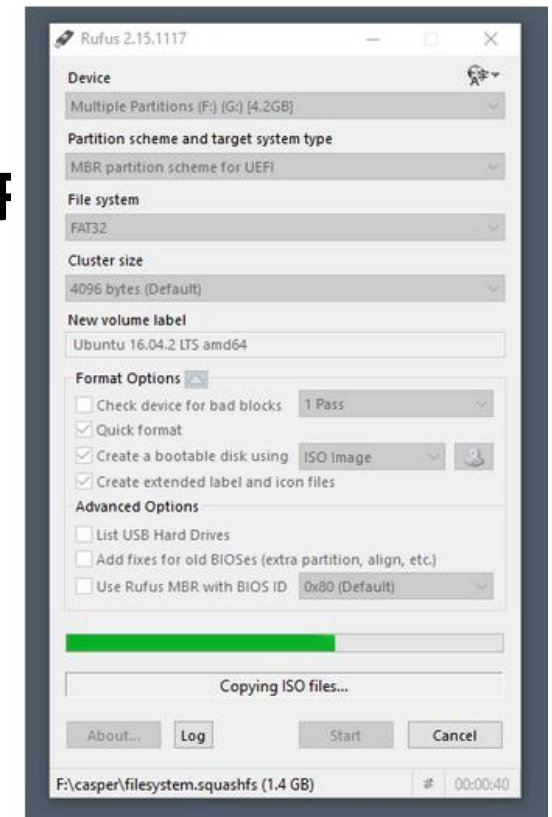
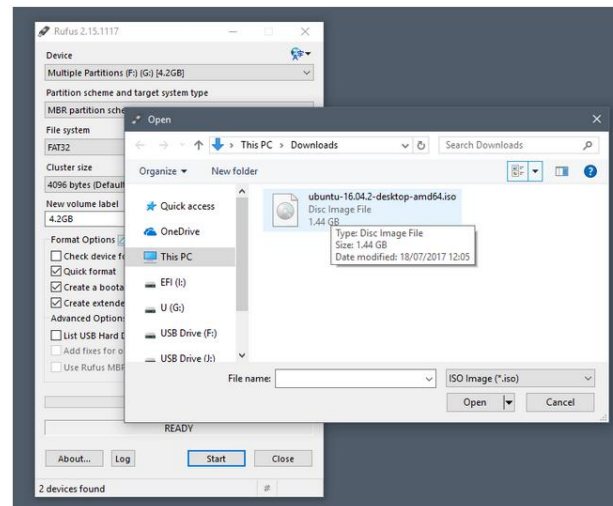
Instalar Ubuntu

- Crear disco de arranque (CD, DVD, USB):
Descarga imagen iso : Ubuntu-20.04.3-desktop-amd.iso



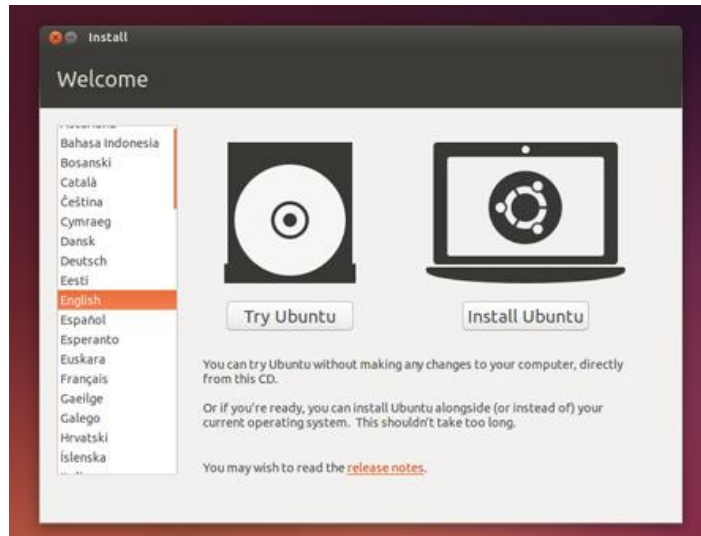
Lanzar Rufus

opción : **MBR partition scheme for UEFI**

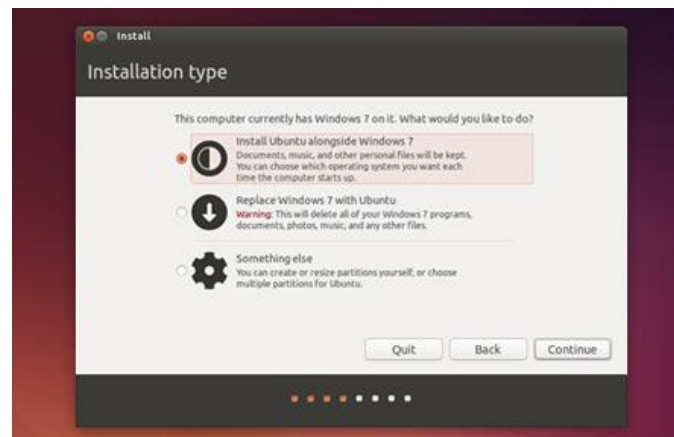


Instalar Ubuntu

- Con la Pc (Laptop) apagada insertar el disco de arranque (CD, DVD, USB).
- Entrar a menú de BIOS y habilitar el arranque desde CD o CDV o USB. Guardar cambios.



- Live session.
- Seguir pasos para instalación



Windows Subsystem for Linux (WSL)

Característica introducida en Windows 10 que nos permite instalar un Kernel Linux directamente sobre el sistema operativo de Microsoft.



Ventajas:

- Fácil de instalar e influye muy poco en el rendimiento general de la PC.
- Ideal para iniciarse en sistemas UNIX.
- Mas rápido que una máquina virtual.

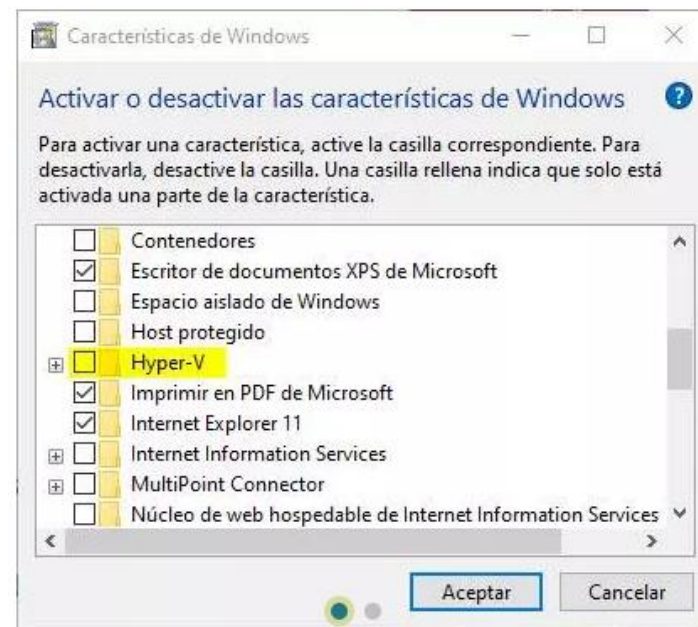
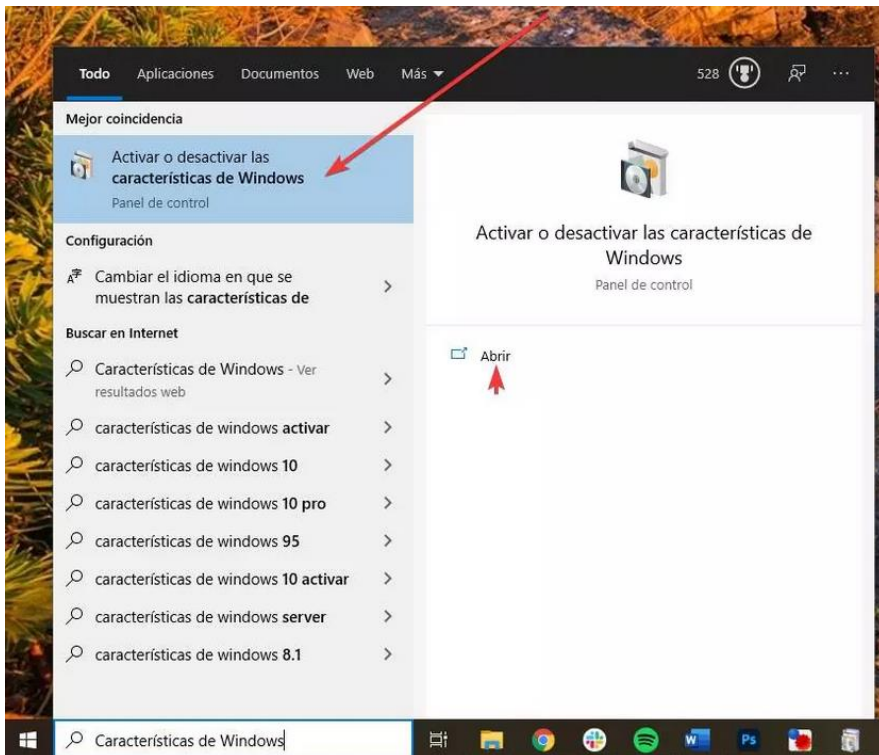
Desventajas:

- Entorno gráfico.
- Consola bash limitada.
- Instalación de paquetes mínima.

Windows Subsystem for Linux



- Requisitos: versión 64-bit de Windows 10 Anniversary Update o superior.
- Activar activar el Subsistema Linux para Windows.



Windows Subsystem for Linux



Probable mensaje de error:

WSL 2 requires an update to its kernel component. For information please visit <https://aka.ms/wsl2kernel>».

Descargar wsl_update_x64.msi de la dirección

https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi

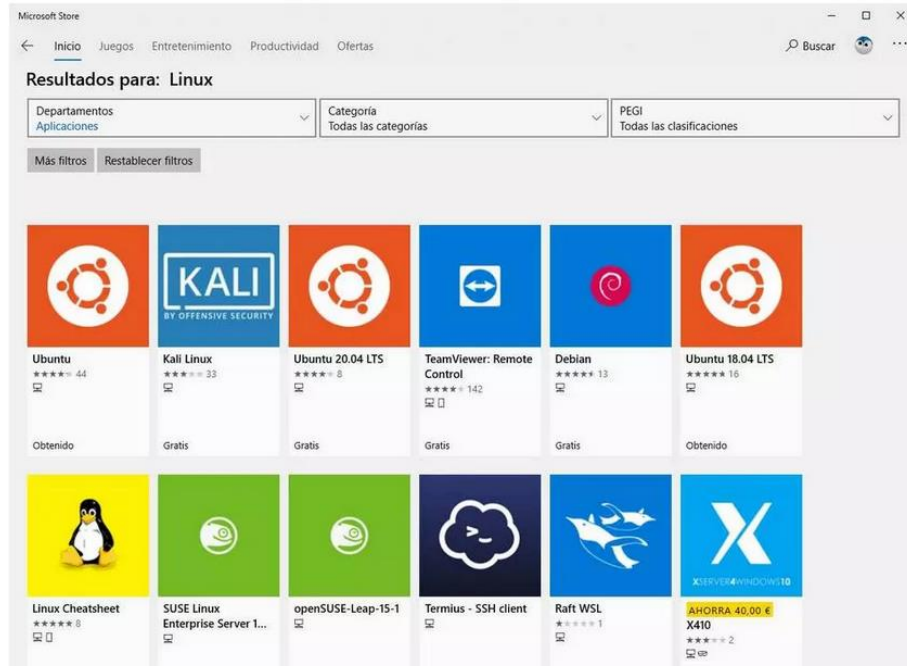
Windows Subsystem for Linux

- En PowerShell



wsr --set-default-version 2

- Instala la distro que más te guste



- Primer comando

sudo apt update && sudo apt upgrade

```
linux@DESKTOP-3H8K53M: $ sudo apt update
Hit:1 http://archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://archive.ubuntu.com/ubuntu xenial-updates InRelease [102 kB]
Get:3 http://archive.ubuntu.com/ubuntu xenial-backports InRelease [102 kB]
Get:4 http://security.ubuntu.com/ubuntu xenial-security InRelease [102 kB]
Get:5 http://archive.ubuntu.com/ubuntu xenial/universe amd64 Packages [7,532 kB]
Get:6 http://archive.ubuntu.com/ubuntu xenial/universe Translation-en [4,354 kB]
Get:7 http://archive.ubuntu.com/ubuntu xenial/multiverse amd64 Packages [144 kB]
Get:8 http://archive.ubuntu.com/ubuntu xenial/multiverse Translation-en [106 kB]
Get:9 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 Packages [553 kB]
Get:10 http://archive.ubuntu.com/ubuntu xenial-updates/main Translation-en [224 kB]
Get:11 http://archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [483 kB]
Get:12 http://archive.ubuntu.com/ubuntu xenial-updates/universe Translation-en [101 kB]
Get:13 http://archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 Packages [8,928 B]
Get:14 http://archive.ubuntu.com/ubuntu xenial-updates/multiverse Translation-en [4,468 B]
Get:15 http://archive.ubuntu.com/ubuntu xenial-backports/main amd64 Packages [4,688 B]
Get:16 http://archive.ubuntu.com/ubuntu xenial-backports/main Translation-en [3,216 B]
Get:17 http://archive.ubuntu.com/ubuntu xenial-backports/universe amd64 Packages [5,624 B]
Get:18 http://archive.ubuntu.com/ubuntu xenial-backports/universe Translation-en [2,872 B]
Get:19 http://security.ubuntu.com/ubuntu xenial-security/main amd64 Packages [284 kB]
Get:20 http://security.ubuntu.com/ubuntu xenial-security/main Translation-en [121 kB]
Get:21 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 Packages [138 kB]
Get:22 http://security.ubuntu.com/ubuntu xenial-security/universe Translation-en [71.5 kB]
Get:23 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 Packages [2,748 B]
Get:24 http://security.ubuntu.com/ubuntu xenial-security/multiverse Translation-en [1,232 B]
Fetched 12.9 MB in 6s (1,949 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
3 packages can be upgraded. Run 'apt list --upgradable' to see them.
linux@DESKTOP-3H8K53M: $
```

Bash



¿Para qué sirve Bash (csh,ksh,tcsh,zsh)?

- Administrar archivos : Listar, acceder, copiar, mover, editar, etc
- Administrar el sistema operativo:
 - Instalación de programas : `sudo apt-get install nombre_programa`
 - compilar, ejecutar programas (Fortran,c++, etc)
 - Modificar variables de entorno:
 - Instalación de librerías para programas.
- Tratar archivos de texto.

Unix/Linux Command Reference

FOSSwire.com

File Commands

ls - directory listing
ls -al - formatted listing with hidden files
cd *dir* - change directory to *dir*
cd - change to home
pwd - show current directory
mkdir *dir* - create a directory *dir*
rm *file* - delete *file*
rm -r *dir* - delete directory *dir*
rm -f *file* - force remove *file*
rm -rf *dir* - force remove directory *dir* *
cp *file1 file2* - copy *file1* to *file2*
cp -r *dir1 dir2* - copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* - rename or move *file1* to *file2*
if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* - create symbolic link *link* to *file*
touch *file* - create or update *file*
cat > *file* - places standard input into *file*
more *file* - output the contents of *file*

System Info

date - show the current date and time
cal - show this month's calendar
uptime - show current uptime
w - display who is online
whoami - who you are logged in as
finger *user* - display information about *user*
uname -a - show kernel information
cat /proc/cpuinfo - cpu information
cat /proc/meminfo - memory information
man *command* - show the manual for *command*
df - show disk usage
du - show directory space usage
free - show memory and swap usage
whereis *app* - show possible locations of *app*
which *app* - show which *app* will be run by default

Compression

tar cf *file.tar files* - create a tar named *file.tar* containing *files*
tar xf *file.tar* - extract the files from *file.tar*

Entrega de trabajos



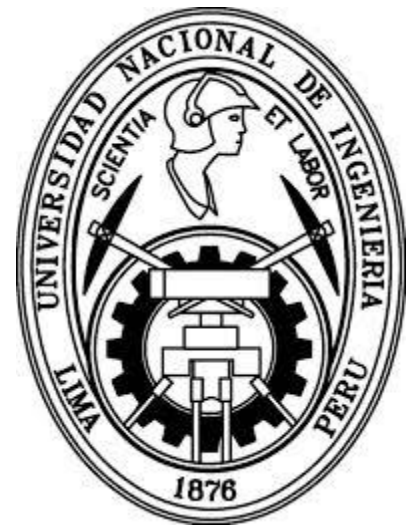
- En cada entrega PC, EP, EF o ES se entrega un solo archivo de texto que contenga el script python.
- El script debe contar con tres partes fundamentales :
 - A) Encabezado : Esta parte es fija (será especificada por el profesor)
 - B) Cuerpo : Implementación del método en particular
 - C) Salida (output): escritura de archivos y producción gráficas (ps,eps o pdf).

Entrega de trabajos



- El script debe estar nombrado : x_y_z.py
x=primer apellido del alumno(a).
y=primer nombre del alumno(a).
z=nombre del trabajo que se entrega (PC1,PC2,PC3,PC4,EP,EF,ES).
- Si no se cumplen las especificaciones se disminuira **5 puntos** sobre la nota obtenida.

Encabezado script python



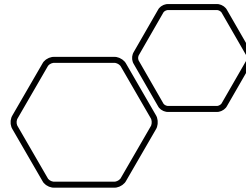
```
#!/usr/bin/python3
```

```
from numpy import *
```

```
import matplotlib.pyplot as plt # from pylab import plot,show
```

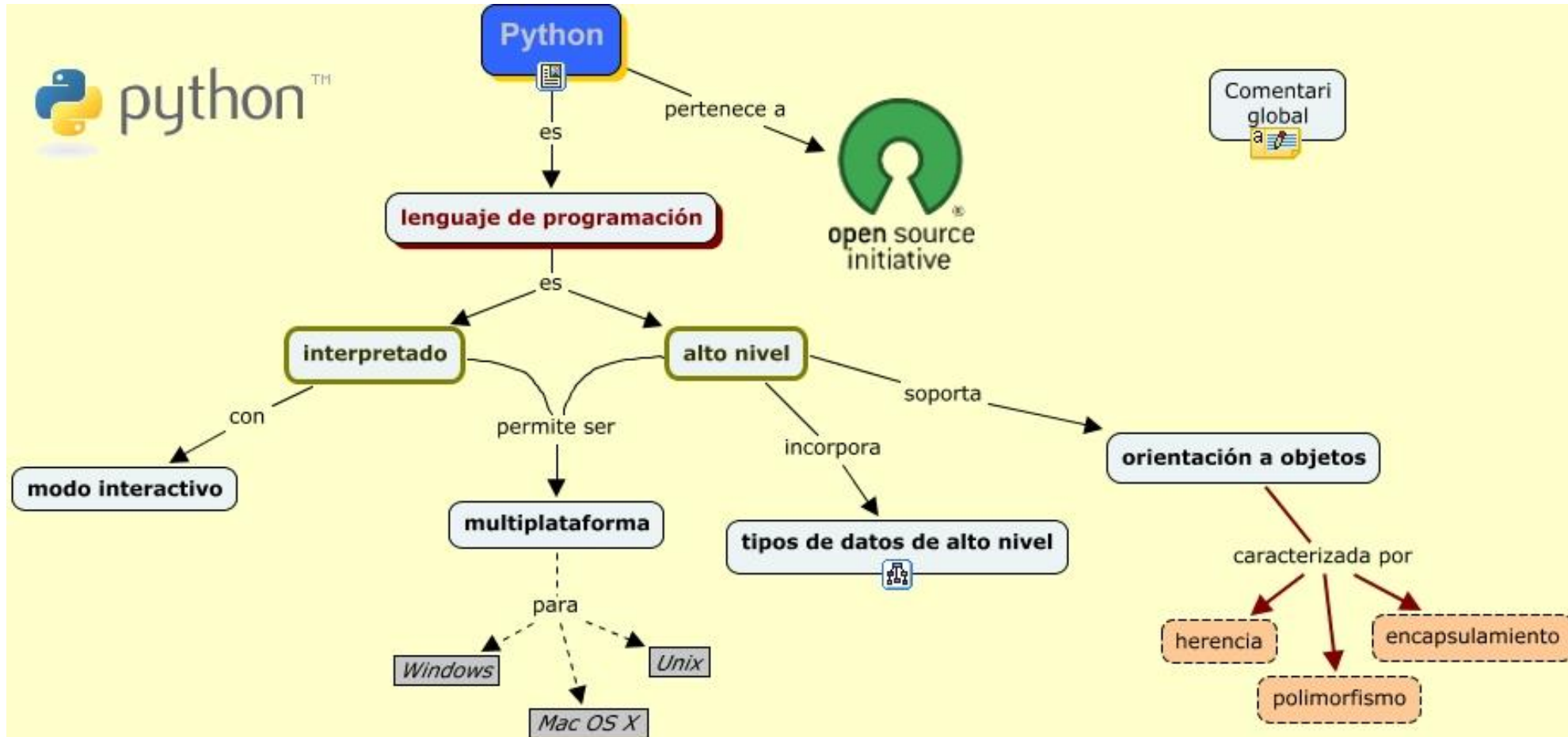
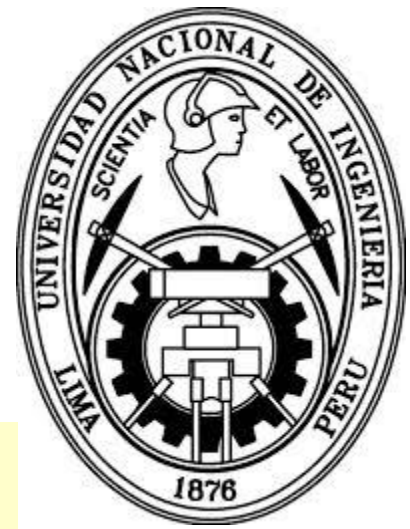
```
import warnings
```

```
warnings.filterwarnings("ignore")
```



Curso introdutorio de Python

Python



Python



- Interprete de comandos (no se necesita compilar).
- Multipropósito: desarrollo web, desarrollo de juegos, desarrollo de software.
- Aprendizaje rápido e intuitivo.
- Python maneja una sintaxis indentada (con márgenes) de carácter **obligatorio**.
- Para separar los bloques de código en Python se debe tabular hacia dentro.
- Python es *case sensitive*: hace diferencia entre mayúsculas y minúsculas.



Instalar Python en Ubuntu Bash

- `sudo apt-get install python3` (Instala python3)
- `sudo apt-get install python-pip python3-pip` (herramienta para inst. paquetes python)
- `sudo pip3 install numpy` (Instala librería numpy de Python)
- `sudo pip3 install matplotlib` (Instala librería matplotlib de Python)

Antes de ejecutar un archivo Python (ejemplo.py) se ejecuta la línea

`chmod +x ejemplo.py`

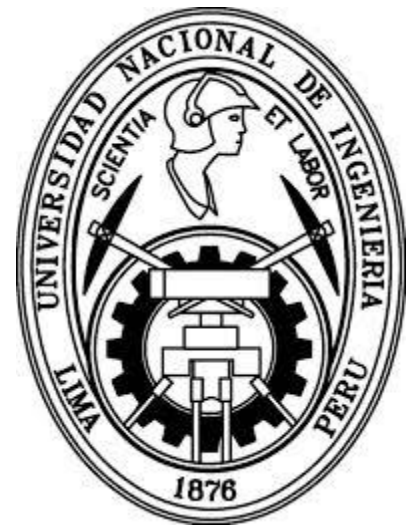
Python: Programación básica



Tipos de variables

- Integer : Son números enteros positivos, negativos o cero. Números como el 1, 0, -2834 son admitidos.
- Float : Son números reales. Números como 3.14.15 ó -6.63×10^{-34} ó 1.0 son admitidos.
- Complex : Números complejos. Números como $1+2j$ ó $-3.5 +0.5j$ son admitidos. Notar que en python la unidad compleja es **j** y no **i**.
- String : almacena letras en forma de una cadena de caracteres.

Python: Programación básica



Asignación de variables

Integer : `x= 1`

Float : `x=1.0`

Float : `x=float (1)`

Complex : `x =1.5 +0j` (variable compleja puramente real)

`x = complex (1.5)` (mismo resultado)

`y= complex (1.0,6.0) → y=1.0 +6.0 j`

String : `x ='esta es un string'`

Python: Programación básica



Output : Imprimir en pantalla

Script_1

```
x=1
y=2
z=2 +3j

print(x)
print ('Este es x=',x,' Este es y= ',y ,'Este es z= ',z)
```

Input : Se entra por teclado el valor de la variable x

Script_2

```
x= input('entra el valor de x: ')
print ('El valor ingresado es',x)
```

Python: Programación básica



Aritmética

Suma : $x+y$

Resta : $x-y$

Multiplicación : $x*y$

División : x/y

Potencia $x**y$ (x elevado a la potencia y)

Ejemplos

$$x+2*y \quad \Leftrightarrow \quad x - 2y$$

$$x - y/2 \quad \Leftrightarrow \quad x - \frac{1}{2}y$$

$$3*x**2 \quad \Leftrightarrow \quad 3x^2$$

$$x/2*y \quad \Leftrightarrow \quad \frac{1}{2}xy$$

Se recomienda el uso de paréntesis

Python: Programación básica



Funciones paquetes y modulos

- Python posee funciones integradas.
- Las funciones se encuentran dentro de los paquetes.
- Cuando los paquetes son muy grandes son divididos en módulos.

`from math import log` (llama al paquete math y da acceso a la función log)

`from math import log,exp,sin,cos,sqrt,pi,e` (llama al paquete math y las función log,exp,etc)

`from math import *` (llama al paquete math y todas las funciones contenidas)

`from numpy.linalg import inv` (llama al paquete numpy, el modulo linalg y la función inv)

Python: Programación básica



Funciones integradas:

log : logaritmo natural

log10: logaritmo base 10

exp : función exponencial.

sin, cos, tan : funciones seno, coseno y tangente. El argumento debe estar en radianes.

asin, acos, atan : funciones seno^{-1} , coseno^{-1} y tangente^{-1} .

sinh, cosh, tanh : funciones senohiperbólico, cosenohiperbólico y tangentehyperbólica.

sqrt : raíz cuadrada

Python: Programación básica



Comentarios

Para poder hacer al Código de fácil lectura para otros usuarios o inclusive para el desarrollador es común comentar los códigos.

Script_3

```
from math import sin,cos,pi
# Ask the user for the values of the radius and angle
r = float(input("Enter r: "))    # radius
d = float(input("Enter theta in degrees: "))
```

Todo texto que se escribe despues del simbolo # es un comentario. Python ignorará cualquier texto despues del simbolo #.



Python: sentencias if y while

La sentencia if

Script_4

```
x = int(input("Enter a whole number no greater than ten: "))
if x>10:
    print("You entered a number greater than ten.")
    print("Let me fix that for you.")
    x = 10
print("Your number is",x)
```

Identación:

- Se debe usar espacios o tabulaciones.
- Se sugiere usar 4 espacios.
- No se recomienda mezclar espacios con tabulaciones.

Python: sentencias if y while



Diferentes condiciones que se pueden utilizar en la sentencia if

if x==1	: verifica si $x = 1$. Note el doble signo igual.
if x>1	: verifica si $x > 1$
if x>=1	: verifica si $x \geq 1$
if x<1	: verifica si $x < 1$
if x<=1	: verifica si $x \leq 1$
if x!=1	: verifica si $x \neq 1$



Python: sentencias if y while

Se pueden utilizar dos condiciones en una sola sentencia

Script_5

Una condición o la otra

```
x = int(input("Enter a whole number: "))  
if x>10 or x<1:  
    print("Your number is either too big or too small.")
```

Una condición y la otra

```
x = int(input("Enter a whole number: "))  
if x<=10 and x>=1:  
    print ("Your number is just right.")
```



Python: sentencias if y while

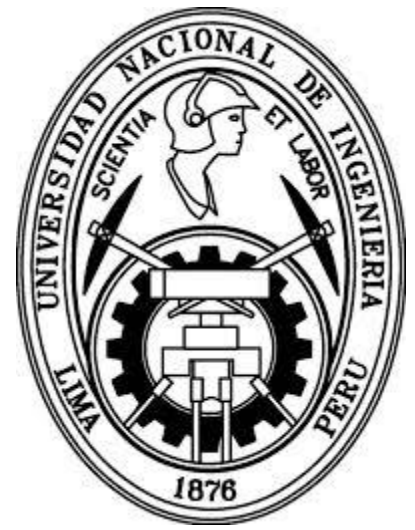
Script_6

Sentencia else

```
if x>10:
    print("Your number is greater than ten.")
else:
    print("Your number is fine. Nothing to see here.")
```

Sentencia elif

```
if x>10:
    print("Your number is greater than ten.")
elif x>9:
    print("Your number is OK, but you're cutting it close.")
else:
    print ("Your number is fine. Move along. ")
```

Python: sentencias if y while

Sentencia while

Las ordenes se ejecutan mientras se verifique la condición ($x > 10$)

Script_7

```
x = int(input("Enter a whole number no greater than ten: "))
while x > 10:
    print("This is greater# than ten. Please try again. ")
    x = int(input("Enter a whole number no greater than ten: "))
print("Your number is", x)
```



Python: Lisas y arreglos

Lista : es una disposición de cantidades una despues de la otra separadas por comas. No es necesario que las cantidades sean del mismo tipo.

Ejemplo

```
r = [ 1, 1, 2, 3, 5, 8, 13, 21 ]
```

```
b = [ 1 , 2 . 5, 3+4. 6 j ]
```

Los elementos de la lista se pueden obtener a partir de expresiones algebraicas.

```
r = [ 2*x, x+y, z/sqrt(X**2+y**2) ]
```



Python: Lisas y arreglos

- Los elementos de la lista son accesibles y están enumerados desde zero.
- Los elementos de la lista se pueden operar y ser argumentos de funciones.

Script_8

```
from math import sqrt
r = [ 1. 0, 1. 5, -2. 2 ]
print(r[0])    # imprime primer elemnto de la lista r
length= sqrt( r[0]**2 + r[1]**2 + r[2]**2)
print(length)
```

Se puede añadir un elemento al final de la lista

```
r = [ 1.0, 1.5, -2.2]
x = 0.8
r.append(2*x+1)
print(r)
```



Python: Lisas y arreglos

Arreglos (arrays) Son muy similares a las listas ya que Tambien es un conjunto ordenado de valores. Sin embargo tienen algunas diferencias:

- El número de elementos de un array es fijo. No se puede anadir elementos.
- Todos los elementos de un array deben ser del mismo tipo. No se puede cambiar el tipo de los elementos una vez creado el array.
- Los arrays pueden ser de dimension dos y entenderse como una matriz. Tambien puden tener mayores dimensiones.
- Se pueden hacer operaciones aritméticas con los arrays.
- Las operaciones con arrays son significativamente más rápidas.



Python: Lisas y arreglos

Para crear un array de dimension 4 lleno de ceros

```
from numpy import zeros  
a = zeros(4,float)  
print(a)
```

Resultado [0. 0. 0. 0.]

Para crear un array de dimension 3X4 lleno de ceros

```
a=zeros([3,4],float)  
print(a)
```

Resultado :[[0. 0. 0. 0. .]
[0. 0. 0. 0.]
[0. 0. 0. 0.]

Script_09



Python: Lisas y arreglos

Creamos un array lleno de ceros y luego asignamos valores

Script_10

```
from numpy import zeros  
a=zeros([2,2],int)  
  
a[0,1]=1  
  
a[1,0] = -1  
print(a)
```

Resultado:

```
[[ 0 1]  
 [-1 0]]
```



Python: Lisas y arreglos

Leer un array desde un archivo de texto

El archivo data.txt contiene lo siguiente

1.0
1.5
-2.2
2.6

Ejecutamos
Script_11

```
from numpy import loadtxt  
a= loadtxt("data.txt",float)  
print(a)
```

Resultado

[1.0 1.5 -2.2 2.6]

Python: Lisas y arreglos



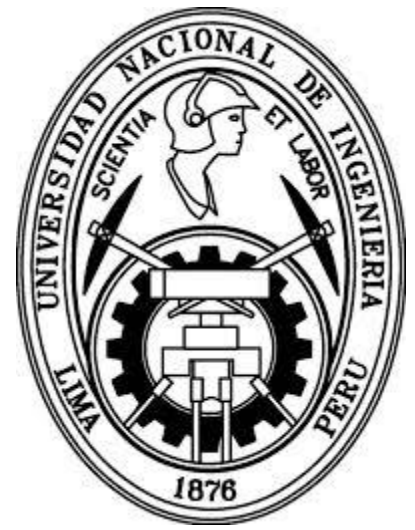
Si data.text contiene

1 2 3 4
3 4 5 6
5 6 7 8

Entonces

```
from numpy import loadtxt  
a= loadtxt("data.txt",float)  
print(a)
```

Imprimirá un array de dimension
3X4



Python: Lisas y arreglos

Las funciones shape y size dan información sobre el tamaño y la forma del array

Script_12

```
a= array([[1,2,3], [4,5,6]] ,int)
print(a.size)
print(a.shape)
```

El resultado será

6

(2, 3)

Python: Lisas y arreglos



Algunos detalles a tener en cuenta con python

Script_13

```
from numpy import array
a= array([1,1] ,int)
b = a
a[0] = 2
print(a)
print(b)
```

Resultado

```
[2 1]
[2 1]
```

```
from numpy import array,copy
a= array([1,1] ,int)
b = copy(a)
a[0] = 2
print(a)
print(b)
```

Resultado

```
[2 1]
[1 1]
```



Python: Bucles for

Un bucle for es un bucle donde el iterador toma ordenadamente valores a partir de una lista.

Script_14

```
r = [ 1, 3, 5]
for n in r:
    print(n)
    print(2*n)
print("Finished")
```

Resultado

1
2
3
6
5
10
Fnished



Python: Bucles for

Se puede crear una lista a partir de una orden y luego usarla en el bucle for

Resultado

Script_15

```
r = range(5)

for n in r:
    print("Hola")
```

Hola
Hola
Hola
Hola
Hola

La función range(5) crea la lista [0,1,2,3,4]



Python: Bucles for

La forma más común
de usar range en un
bucle for

Script_16

```
for n in range(5):  
    print(n**2)
```

Resultado

0

1

4

9

16

Variantes de range

range(5) da [0, 1, 2, 3, 4]

range(2,8) da [2, 3, 4, 5, 6, 7]

range(2,20,3) da [2, 5, 8, 11, 14, 17]

range(20,2,-3) da [20, 17, 14, 11, 8, 5]

Python: Funciones definidas por usuario



Se pueden definir funciones que den como resultado un número

El resultado de una función también puede ser un array

Script_17

```
def factorial(n):  
    f = 1.0  
    for k in range(1,n+1):  
        f *= k  
    return f
```

```
def cartesian(r,theta):  
    x = r*cos(theta)  
    y = r*sin(theta)  
    position = [x,y]  
    return position
```

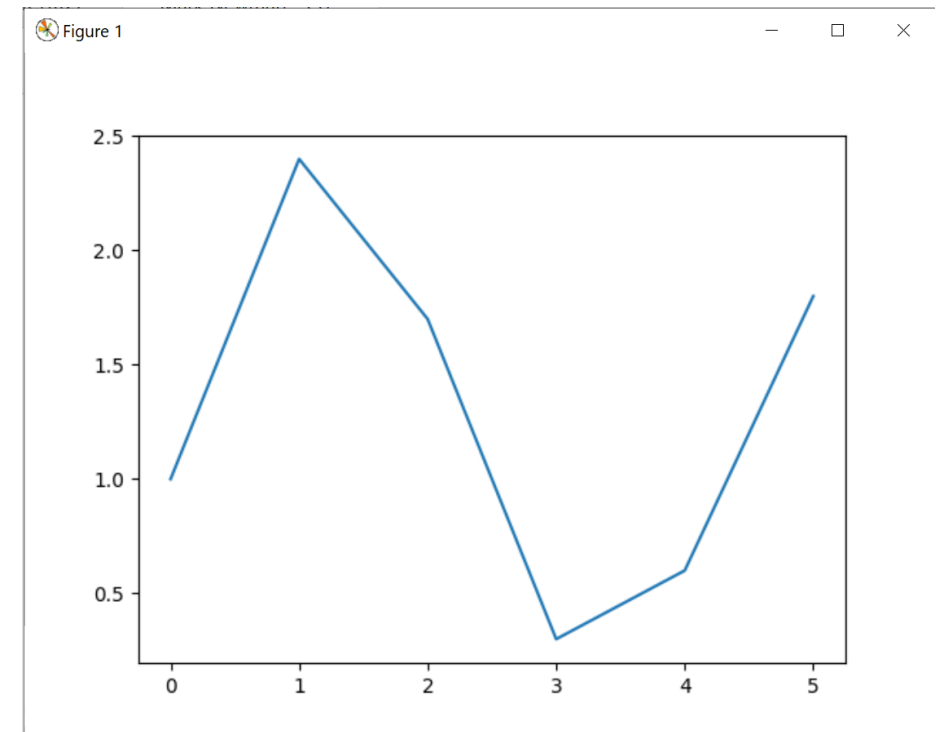
Python: Graficas y visualización



Para graficar se llaman a las funciones plot y show del paquete pylab

Script_18

```
from pylab import plot, show
y = [ 1.0, 2.4, 1.7, 0.3, 0.6, 1.8]
plot (y)
show()
```

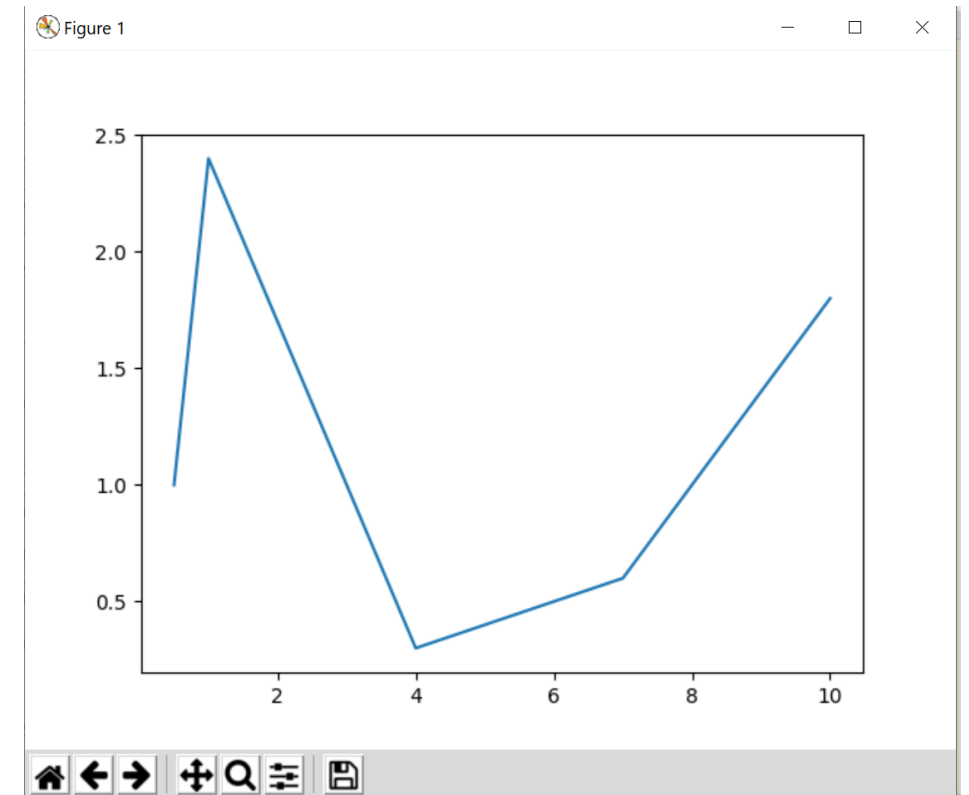


Python: Graficas y visualización



Se puede definir los puntos de las abscisas y ordenadas

```
from pylab import plot,show
x = [ 0. 5, 1. 0, 2. 0, 4. 0, 7. 0, 10. 0 ]
y = [ 1.0, 2.4, 1.7, 0.3, 0.6, 1.8]
plot(x,y)
show()
```



Python: Graficas y visualización



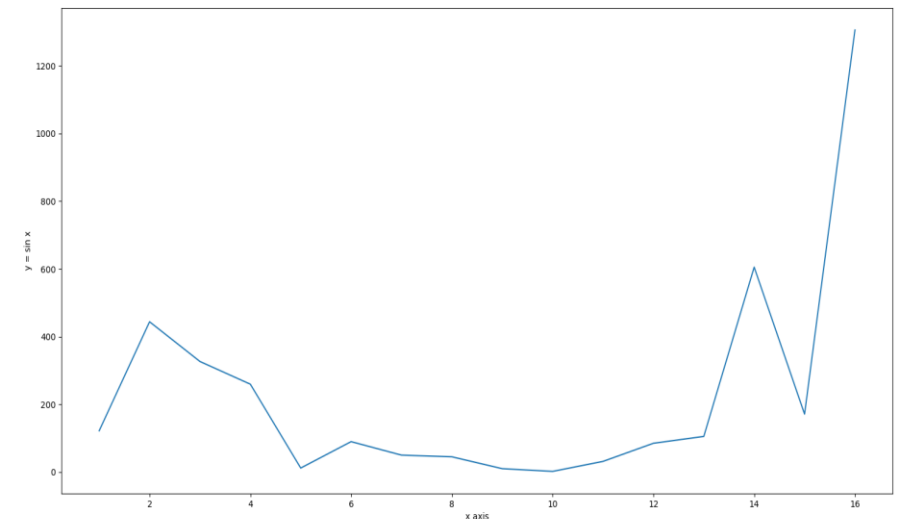
Graficar el archivo plot.text

Ejecutamos

```
1 122.189988
2 444.0270889
3 326.7205784
4 259.7201836
5 12.15722275
6 90.15900043
7 1005.310014
```

.....

```
from numpy import loadtxt
from pylab import plot, show
data= loadtxt("plot.text",float)
x = data[:,0]
y = data[:,1]
xlabel("x axis")
ylabel("y = sin x")
plot(x,y)
show ()
```



Modulos Python



- Son archivos Python con la extensión .py. Ejemplo mi_modulo.py
- Implementan un conjunto de funciones.
- Son importados desde otros módulos usando el comando **import**.
- Se recomienda que genere un módulo por cada tema abordado en el curso.

Modulos Python



```
#!/usr/bin/python3
from numpy import *
#####
from math import *
#####
import warnings
warnings.filterwarnings("ignore")
#####
def factorial(n):
    f = 1.0
    for k in range(1,n+1):
        f *= k
    return f

def multiplication(num1,num2):

    return num1*num2
```

```
#!/usr/bin/python3
from numpy import *
#####
from math import *
#####
import warnings
warnings.filterwarnings("ignore")
#####
from mi_modulo import *
#####

a=multiplication(3,2)

print(factorial(3),a)
```

Buenas prácticas de programación



- Hacer comentarios.
- No usar funciones que hagan un trabajo muy particular.
- Los nombres de las variables deben tener relación con el dato que representan.
- Mantener un script limpio, ordenado y mínimo.
- No invertir mucho tiempo en la optimización del script. El primer objetivo es que funione !!!

Mapa logístico de Feigenbaum (1944-2019)



Es un mapeo a menudo citado como un ejemplo de comportamiento caótico puede surgir en ecuaciones dinámicas no lineales muy simples. El mapa fue popularizado en un artículo de 1976 por el biólogo Robert May, en parte como un modelo demográfico de tiempo discreto. Matemáticamente, el mapa logístico está descrito por

$$x_{n+1} = r x_n (1 - x_n)$$

donde r es una constante.



Mapa logístico de Feigenbaum (1944-2019)

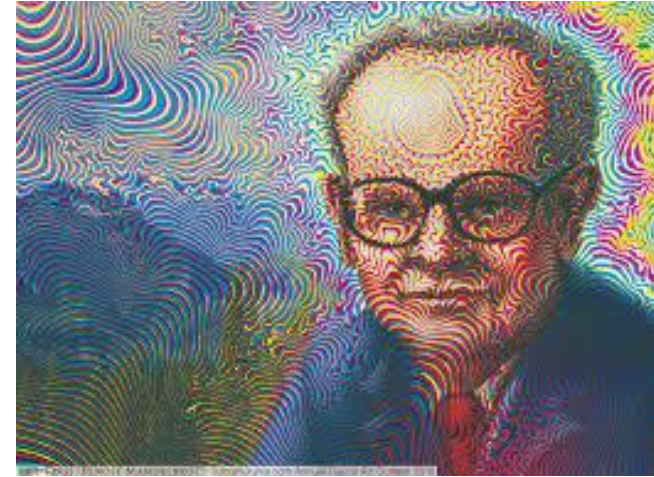


- Para un fijo $r=1$ y un valor inicial $x_0 = 0.45, 0.5, 0.55$, graficar la evolución x_{k+1} versus k (iteración). Considere 1000 iteraciones. Las tres gráficas deben estar en una sola imagen.
- Realizar la gráfica (r, x) donde $r \in [1, 4]$ incrementandose en 0.1. Debe tomar como valor inicial $x_0 = 0.5$ en todos los casos y solo graficar los valores de x_{k+1} . En cada caso se deben realizar 3000 iteraciones pero solo deben graficarse los valores de x_{k+1} a partir de la iteración 1000.

El conjunto de Mandelbrot



Considere la ecuación $z' = z^2 + c$,
donde z es un número complejo y c es un
número complejo constante. Para cualquier
valor dado de c , la ecuación transforma un
numero z en un complejo.



Para un número complejo dado de c , comience con $z = 0$ obtenga z' . Reinserte z' en el lugar de z y repita sucesivamente. Si la magnitud $|z|$ del valor resultante es en algún caso mayor que 2, entonces el punto en el plano complejo en la posición c no está en el conjunto de Mandelbrot; de lo contrario, está en el conjunto.

El conjunto de Mandelbrot



- Considere los números complejos $c=x+yi$ en una malla de $N \times N$ puntos ($N=100$) dentro de la región $[-2,2] \times [-2,2]$. Dibuje los puntos de la malla coloreando de color negro los puntos que pertenecen al conjunto de Mandelbrot y de color blanco los que no pertenecen.
- Para el análisis de cada punto realice 100 iteraciones.
- Incremente el valor N según su criterio para mejorar la calidad la imagen.
- Incremente el número de iteraciones.