Repaso de Python

1) Opciones de Jupyter Notebook

Opciones de Celdas

- Code
- Markdown
- Heading
- Raw NBConvert

Shortcuts Importantes

- Shift + Enter: Permite ejecutar la celda actual
- Ctrl + Enter: Ejecuta las celdas seleccionadas
- Alt + Enter: Ejecuta la celda actual, e inserta una nueva celda debajo de la celda actual
- A: Inserta una nueva celda encima de la celda actual
- B: Inserta una nueva celda debajo de la celda actual
- D (presionar dos veces): Elimina la celda
- H: Muestra todos los shortcuts

**Para ejecutar los shortcuts no se debe de tener la celda seleccionada*

2) Repaso Python

Comentarios

In [5]:
```
# La siguiente función genera números en un rango determinado
range(10)
```

Out[5]:
```
range(0, 10)
```

Imprimir datos por pantalla

In [6]:
```
# La siguiente función realizara un print de Hola Mundo
print("Hola Mundo")
```

```
Hola Mundo
```

In [8]:
```
print("Hola Mundo","De Nuevo")
```

```
Hola Mundo De Nuevo
```

Parametros de la función print

Parametro	Descripción
sep='separator'	Permite especificar el separador utilizado, el default es ' '
end='end'	Permite especificar el valor final utilizado, el default es '\n'
file	Permite escribir el objeto. Por default es sys.stdout pero se puede especificar un archivo
flush	Booleano, permite especificar si el buffer interno es eliminado o no

In [9]:
```
# Parametros sep
print("Hola Mundo","De Nuevo",sep="-")
```

```
Hola Mundo-De Nuevo
```

In [10]:
```
# Parametros sep
print("Hola Mundo","De Nuevo",end="!!")
```

```
Hola Mundo De Nuevo!!
```

In [12]:
```
# Parametro file
prueba = open('prueba.txt', 'w')

print('Hola Mundo', file = prueba)
prueba.close()
```

In [13]:
```
# Parametro flush
import time
print('Cargando ', end='',flush=True)
for i in range(0,5):
    time.sleep(1)
    print('.',end='',flush=True)
```

```
Cargando .....
```

Lecturade datos por teclado

In [14]:
```
input("Ingrese un número: ")
```

```
Ingrese un número: 3
```

Out[14]:
```
'3'
```

Variables y Tipos de Datos

In [15]:
```
edad = 30
type(edad)
```

Out[15]:
```
int
```

In [16]:
```
talla = 1.80
type(talla)
```

Out[16]:
```
float
```

In [47]:
```
x=1j
type(x)
```

Out[47]:
```
complex
```

In [17]:
```
nombre= 'Alexis A. Quezada C.'
type(nombre)
```

Out[17]:
```
str
```

In [43]:
```
condicion= True
type(condicion)
```

Out[43]:
```
bool
```

In [48]:
```
lista = ["perro","gato"]
type(lista)
```

Out[48]:
```
list
```

In [50]:
```
tupla = ("perro","gato")
type(tupla)
```

Out[50]:
```
tuple
```

In [51]:
```
x= range(10)
type(x)
```

Out[51]:
```
range
```

In [52]:
```
x= {"perro","gato"}
type(x)
```

Out[52]:
```
set
```

In [54]:
```
x = dict(nombre="Alexis", edad=30)
type(x)
```

Out[54]:
```
dict
```

In [58]:
```
x = set(("perro","gato"))
type(x)
```

Out[58]:
```
set
```

Casting

In [40]:
```
x = 30
x = float(x)
x
```

Out[40]:
```
30.0
```

In [41]:
```
x = "30"
x = float(x)
x
```

Out[41]:
```
30.0
```

In [42]:
```
x=str(1)
x
```

Out[42]:
```
'1'
```

Strings

In [60]:
```
#String multilinea
a = """Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua."""
print(a)
```

```
Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.
```

In [66]:
```
#Iterando a traves de string
texto = "Hola"
for letra in texto:
    print(letra)
```

```
H
o
l
a
```

In [67]:
```
#Obteniendo Longitud de String
a = "Hola Mundo"
len(a)
```

Out[67]:
```
10
```

In [69]:
```
#Verificando si frase o caracter se encuentra en string
a = 'Hola Mundo'
print("Hola" in a)
```

```
True
```

Cortando Strings

In [64]:

```
# Obteniendo Substring
a = "Hola Mundo"
print(a[2:5])
```

```
la
```

In [70]:
```
#Cortando string del inicio hasta una posición x
a = "Hola Mundo"
a[:2]
```

Out[70]:
```
'Ho'
```

In [71]:
```
#Cortando string de una posición x al final
a = "Hola Mundo"
a[2:]
```

Out[71]:
```
'la Mundo'
```

In [75]:
```
#Invierte String
a[::-1]
```

Out[75]:
```
'odnuM aloH'
```

Modificando String

In [81]:
```
#Convirtiendo a Mayusculas
a.upper()
```

Out[81]:
```
'HOLA MUNDO'
```

In [82]:
```
#Convirtiendo a Minisculas
a.lower()
```

Out[82]:
```
'hola mundo'
```

In [84]:
```
#Reemplaza caracter especifico por otro
b=a
b.replace("H","J")
```

Out[84]:
```
'Jola Mundo'
```

In [85]:
```
#Separa string en substrings si el separador se encuentra
b=a
b.split(" ")
```

Out[85]:
```
['Hola', 'Mundo']
```

Format Strings

In [86]:
```
numero = 20
txt = "Quiero sacar un {}"
print(txt.format(numero))
```

```
Quiero sacar un 20
```

Caracteres Especiales

Code	Result
\'	Comilla Simple
\\	Backslash
\n	Nueva Linea

Operaciones aritmeticas

Operador	Nombre	Ejemplo
+	Suma	x + y
-	Resta	x - y
*	Multiplicación	x * y
/	División	x / y
%	Módulo	x % y
**	Exponenciación	x ** y
//	División de piso	x // y

In [23]:
```
# Modulo
10%3
```

Out[23]: 1

In [25]:
```
# Exponenciación
2**3
```

Out[25]: 8

In [26]:
```
# División de Piso
10//3
```

Out[26]: 3

In [27]:
```
10/3
```

Out[27]: 3.3333333333333335

In [35]:
```
nombre="Alexis"
apellido = "Quezada"
nombre+" "+apellido
```

Out[35]: 'Alexis Quezada'

In [33]:
```
nombre *2
```

Out[33]: 'Alexis Alexis '

Operadores Lógicos

Operador	Descripción	Ejemplo
and	Regresa True si ambos clausulas son verdaderas, caso contrario Regresa False	x < 5 and x < 10
or	Regresa True si al menos uno es verdadero	x < 5 or x < 4
not	Regresa el resultado opuesto, regresa False si es True	not(x < 5 and x < 10)

In [29]:
```
2 < 5 and  9 < 10
```

Out[29]:
```
True
```

In [30]:
```
1 < 5 or 5 < 4
```

Out[30]:
```
True
```

In [31]:
```
not(2 < 5 and  9 < 10)
```

Out[31]:
```
False
```

Colecciones

Existen 4 tipos de Colecciones:

1. List: Es una coleccion que esta ordenado y es mutable. Permite valores duplicados
2. Tupla: Es una coleccion que esta ordenado y no es mutable. Permite valores duplicados
3. Set: Es una coleccion que no esta ordenado,es mutable, y no indexada. No permite valores duplicados
4. Dictionary: Es una coleccion que esta ordenado y es mutable. Permite valores duplicados

Listas

In [91]:
```
#Obteniendo valores x en lista
lista = [2,'tres',4]
lista[2]
```

Out[91]:
```
4
```

In [95]:
```
#Obteniendo valores en lista con indice negativo
lista[-1]
```

Out[95]:
```
4
```

In [98]:
```
#Obteniendo rango de valores
lista[0:2]
```

Out[98]:
```
[2, 'tres']
```

In [101…
```
#Obteniendo rango de valores desde el inicio
lista[:2]
```

Out[101…
```
[2, 'tres']
```

In [104…
```
#Obteniendo rango de valores desde posicion indicada hasta final
lista[1:]
```

Out[104…
```
['tres', 4]
```

In [105…
```
#Agregando valores a lista
lista.append("cinco")
lista
```

Out[105…
```
[2, 'tres', 4, 'cinco']
```

In [106…
```
#Agregando valores a lista en posicion especifica
lista.insert(0,1)
lista
```

Out[106…
```
[1, 2, 'tres', 4, 'cinco']
```

In [110…
```
lista1=["manzana","platano"]
lista2=["mango","papaya"]
```

In [111…
```
#Agregando elemento de una lista a otra lista
lista1.extend(lista2)
lista1
```

Out[111…
```
['manzana', 'platano', 'mango', 'papaya']
```

In [112…
```
lista1=["manzana","platano"]
lista2=["mango","papaya"]
#Agregando elemento de una lista a otra lista
lista3=lista1+lista2
lista3
```

Out[112…
```
['manzana', 'platano', 'mango', 'papaya']
```

In [113…
```
#Quita elemento especifico
lista3.remove("mango")
lista3
```

Out[113…
```
['manzana', 'platano', 'papaya']
```

In [115…
```
#Quita elemento en posicion especifica
lista3.pop(2)
lista3
```

Out[115…
```
['manzana', 'platano']
```

In [118…
```
#Cuenta las ocurrencias de elementos en la lista
lista3.count('manzana')
```

Out[118…
```
1
```

Sets

In [124…
```
#Union de sets
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}

set3 = set1.union(set2)
print(set3)
```

```
{1, 'c', 2, 3, 'b', 'a'}
```

In [127…
```
#Interseccion de sets
set1 = {1,2,3}
set2 = {1,4,5}
set3 = set1.intersection(set2)
set3
```

Out[127…
```
{1}
```

In [128…
```
#diferencia simetrica de sets
set1 = {1,2,3}
set2 = {1,4,5}
set3 = set1.symmetric_difference(set2)
set3
```

Out[128…
```
{2, 3, 4, 5}
```

Diccionarios

In [129…
```
diccionario = {
  "marca": "Chevrolet",
  "modelo": "Cobalt",
  "anio": 2010
}
print(diccionario)
```

```
{'marca': 'Chevrolet', 'modelo': 'Cobalt', 'anio': 2010}
```

In [130…
```
diccionario2 = dict(nombre = "Alexis Quezada", edad = 30, pais = "Peru")
diccionario2
```

Out[130…
```
{'nombre': 'Alexis Quezada', 'edad': 30, 'pais': 'Peru'}
```

In [132…
```
#Obteniendo valor asociado a llave del diccionario
diccionario["modelo"]
```

Out[132…
```
'Cobalt'
```

In [133…
```
diccionario.get("modelo")
```

Out[133…
```
'Cobalt'
```

In [134…
```
#Obtener las llaves del diccionario
diccionario2.keys()
```

Out[134… dict_keys(['nombre', 'edad', 'pais'])

In [135…
```
#Obtener los valores del diccionario
diccionario2.values()
```

Out[135… dict_values(['Alexis Quezada', 30, 'Peru'])

In [138…
```
#Diccionario anidado
diccionario3 = {
     "elemento1": {
      "marca": "Chevrolet",
      "modelo": "Cobalt",
      "anio": 2010
    },
    "elemento2": {
      "marca": "Ford",
      "modelo": "Mustang",
      "anio": 1998
    }
}
print(diccionario3)
```

```
{'elemento1': {'marca': 'Chevrolet', 'modelo': 'Cobalt', 'anio': 2010}, 'elemento2': {'mar
ca': 'Ford', 'modelo': 'Mustang', 'anio': 1998}}
```

In [140…
```
diccionario3["elemento1"]["marca"]
```

Out[140… 'Chevrolet'

In [141…
```
len(diccionario3)
```

Out[141… 2

Condicionales

In [142…
```
a = 200
b = 33
if b > a:
  print("b mayor que a")
elif a == b:
  print("a y b son iguales")
else:
  print("a es mayor que b")
```

```
a es mayor que b
```

Loops

While Loops

In [147…
```
i=0
while i<4:
    print(i)
    i+=1
```

```
0
1
```

```
2
3
```

In [148…
```
#la clausula break nos permite salir del loop incluso si la condicion del while sigue sier
i=0
while i<4:
    print(i)
    if i==2:
        break
    i+=1
```

```
0
1
2
```

For Loops

In [151…
```
for i in range(4):
    print(i)
```

```
0
1
2
3
```

In [153…
```
#la clausula continue nos permite parar la iteracion actual y continuar con la siguiente
for i in range(4):
    if i==1:
        continue
    print(i)
```

```
0
2
3
```

In [3]:
```
# One liner para for loop
x = [1, 2, 3, 4, 5]
y = [2*a for a in x if a % 2 == 1]
print(y)
```

```
[2, 6, 10]
```

In [6]:
```
# La siguiente función ejecuto lo mismo que la anterior función
y=[]
for a in x:
    if a%2 ==1:
        y.append(2*a)
print(y)
```

```
[2, 6, 10]
```

Funciones

In [157…
```
def funcion(nombre,apellido):
    print(apellido,nombre,sep=", ")
```

In [159…
```
funcion("Alexis","Quezada")
```

```
Quezada, Alexis
```

Argumentos Arbitrarios (*args)

Si no se sabe cuatos argumentos van a ser pasados la funcion, se añade un * antes del nombre de parametro

In [169…
```
def cursos(*curso):
    cur=''
    for c in curso:
        if cur!="":
            cur+=","+c
        else:
            cur+=c
    return print("Los cursos que llevan los alumnos son: " + cur)
```

In [170…
```
cursos("Big Data", "IA", "Base de Datos")
```

```
Los cursos que llevan los alumnos son: Big Data,IA,Base de Datos
```

Argumentos Clave Arbitrarios (*kwargs)

Si no se sabe cuatos argumentos van a ser pasados la funcion y se acepta argumetnos nombrados, se añade un ** antes del nombre de parametro

In [171…
```
def concatenar(**kwargs):
    resultado = ""
    for arg in kwargs.values():
        resultado += arg
    return resultado

print(concatenar(a="Python", b="es", c="Genial", d="!"))
```

```
PythonesGenial!
```

Importante

si se usan los 3 tipos de argumentos este debe ser el orden
func(fargs, *args, **kwargs)

In [73]:
```
def funcionArgumentos(arg1,*args,**kwargs):
    print("Argumento 1 ",arg1)
    print("*args ",args)
    print("**kwargs",kwargs)

funcionArgumentos("probando",1,2,3,clave1='123',clave2='345')
```

```
Argumento 1  probando
*args  (1, 2, 3)
**kwargs {'clave1': '123', 'clave2': '345'}
```

Funciones Lambda / Funcion Anonima

In [173…
```
x = lambda a : a + 10
print(x(5))
```

```
15
```

In [174…
```
x = lambda a, b : a * b
print(x(5, 6))
```

```
30
```

In [176…
```
def myfunc(n):
  return lambda a : a * n

mydoubler = myfunc(2)

print(mydoubler(11))
```

```
22
```

In [2]:
```
serie = [23,45,57,39,1,3,95,3,8,85]
resultado = filter (lambda m: m > 29, serie)
print('Regresa todo los numero en la serie mayores a 29 :',list(resultado))
```

```
Regresa todo los numero en la serie mayores a 29 : [45, 57, 39, 95, 85]
```

Clases y Objetos

Python es un Programa Orientado a Objetos (POO)
Casi todo en python es un objeto, con propiedades y metodos.
Una clase es como un constructor de objetos o plano para crear objetos.

In [1]:
```
class Persona:
    def __init__(self,nombre,edad):
        self.nombre = nombre
        self.edad = edad

p1 = Persona("Alexis",30)
print(p1.nombre)
```

```
Alexis
```

Como se puede observar del ejemplo anterior, se utilizo una función **init**, la cual es una función incorporada la cual funciona como constructor cada vez que un objeto es creado desde una clase.

Otra funcion incoporada es el **str**, la cual controla como una clase objeto es representada como un string.

In [8]:
```
#Ejemplo de como regresa la clase si no se especifica la funcion __str__
print(p1)
```

```
<__main__.Persona object at 0x000001C4EFA78DF0>
```

In [10]:
```
#Ejemplo de como regresa la clase si se especifica la funcion __str__
class Persona:
    def __init__(self,nombre,edad):
        self.nombre = nombre
        self.edad = edad

    def __str__(self):
        return f"{self.nombre} tiene {self.edad}"
p1 = Persona("Alexis",30)
print(p1)
```

```
Alexis tiene 30
```

Metodos de Objetos

Los objetos pueden tener metodos. Los metodos en los objetos son funciones que pertenecen al objeto.

In [7]:
```
#Creamos el metodo edadFuturo, que espere un valor adicional
class Persona:
    def __init__(self,nombre,edad):
        self.nombre = nombre
        self.edad = edad

    def __str__(self):
        return f"{self.nombre} {self.edad}"

    def edadFutura(self,anio):
        print("De aca a  " + str(anio) + " años, " + self.nombre + " tendra " + str(self.e

p1 = Persona("Alexis",30)
p1.edadFutura(5)
```

```
De aca a  5 años, Alexis tendra 35
```

Herencia

En la herencia existen los dos tipos de clase padre e hijo

In [3]:
```
#Crearemos una clase hijo en base a la clase persona
class Estudiante(Persona):
    pass
```

In [11]:
```
x=Estudiante("Alex",20)
print(x)
```

```
Alex 20
```

In [12]:
```
x.edadFutura(10)
```

```
De aca a  10 años, Alex tendra 30
```

También podemos especificar que herede todas las propiedades de la clase padre de la siguiente manera

In [21]:
```
#Utilizamos el constructor __init__ para heredar los metodos de la clase padre
class Estudiante(Persona):
    def __init__(self,nombre,edad):
        Persona.__init__(self,nombre,edad)

x=Estudiante("Alex",21)
print(x)
```

```
Alex 21
```

In [24]:
```
#Podemos utilizar de igual manera la funcion super() para que la clase hijo here del padre

class Estudiante(Persona):
    def __init__(self,nombre,edad):
        super().__init__(nombre,edad)

x=Estudiante("Alex",20)
print(x)
```

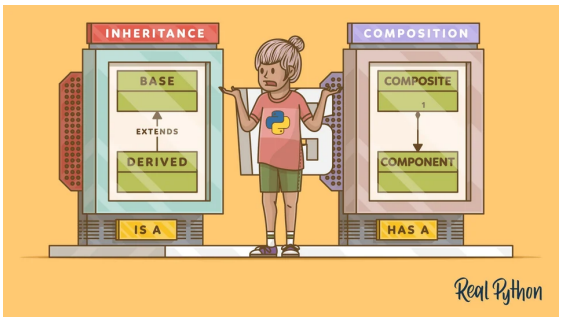
```
Alex 20
```

También podemos añadir metodos a la clase hijo

In [26]:
```
class Estudiante(Persona):
    def __init__(self,nombre,edad, egresa):
        super().__init__(nombre,edad)
        self.graduacion = egresa
    def promocion(self):
        print(" Alumno {0} de la promocion {1}".format(self.nombre,self.graduacion))
x=Estudiante("Alex",20,202301)
x.promocion()
```

```
 Alumno Alex de la promocion 202301
```

Composición



Como vimos anteriormente, la **herencia** es cuando una clase se deriva de una clase base o padre.
Un ejemplo de esto fue la clase Persona, que tenia ciertos elementos que hemos podido heredar con la clase Estudiante, ya que es un subtipo, es decir un estudiante es una persona.

En una relación de herencia:

- *Clases que heredan de otra clase son llamadas clases derivadas, subclases, o subtipos*
- *Clases de las cuales otras clases son derivadas, son llamadas clase base o super clase*

Ahora una **Composición** nos permite crear tipos complejos creando objetos de otros tipos. Esto quiere decir que una clase compuesta contiene un objeto otro componente de clase

In [63]:
```
class Salario:
    def __init__(self,salario,comisiones):
        self.salario=salario
        self.comisiones = comisiones
    def asignarSalarioBase(self):
        print(self.salario)
class Empleado:
    def __init__(self,salario,comisiones):
        self.salario_base=Salario(salario, comisiones).asignarSalarioBase()
    def salarioIT(self):
        return self.salario_base+500
emp=Empleado(500,0)
```

```
500
```

In [99]:
```
class Salario:
    def __init__(self,**kwargs):
        self.salario=kwargs["salario_base"]
        if 'comisiones' in kwargs:
            self.comisiones=kwargs["comisiones"]
    def asignarSalarioBase(self):
        return self.salario
```

```
class Empleado:
    def __init__(self,salario,comisiones):
        self.salario_base=Salario(salario_base=salario,comisiones=comisiones).asignarSalar
        self.comisiones= comisiones
    def salarioSinComisiones(self):
        return self.salario_base+500
    def salarioConComision(self):
        return self.salario_base+self.comisiones

emp=Empleado(930,0)
print("Empleado 1 gana " + str(emp.salarioSinComisiones()))
emp1=Empleado(1200,700)
print("Empleado 2 gana " + str(emp1.salarioConComision()))
```

```
Empleado 1 gana 1430
Empleado 2 gana 1900
```

Manejo de Errores

Para el manejo de errores vamos a utilizar el Try Except

In [101…

```
#Generamos el siguiente error para mostrar como pasa por la excepción, y el código pueda s
try:
    print(z)
except:
    print("Ocurrio un error")
```

```
Ocurrio un error
```

In [103…

```
try:
    print(z)
except:
    print("Ocurrio un error")
finally:
    print("Luego del try except")
```

```
Ocurrio un error
Luego del try except
```

In [104…

```
#Generamos el siguiente error para mostrar como pasa por la excepción, y el código pueda s
z=1
try:
    print(z)
except:
    print("Ocurrio un error")
finally:
    print("Luego del try except")
```

```
1
Luego del try except
```

In [105…

```
z=1
try:
    print(z)
except:
    print("Ocurrio un error")
else:
    print("No ocurrio ningun error")
```

```
1
No ocurrio ningun error
```

Raise un excepción

Podemos utilizar raise cuando queremos levantar una excepción dado una condición especfica

In [107…
```
z=-1
if z<0:
    raise Exception("No se aceptan valores menor a 0")
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_34104/3810704436.py in <module>
      1 z=-1
      2 if z<0:
----> 3     raise Exception("No se aceptan valores menor a 0")

Exception: No se aceptan valores menor a 0
```

In [108…
```
z=-1
if not type(x) is str:
    raise TypeError("No se aceptan ints")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_34104/3666881124.py in <module>
      1 z=-1
      2 if not type(x) is str:
----> 3     raise TypeError("No se aceptan ints")

TypeError: No se aceptan ints
```

In [ ]:
```

```

Assert

Para el debug también podemos usar Assert

In [109…
```
x=11
assert x%2==0, "El numero debe ser par"
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_34104/1574953748.py in <module>
      1 x=11
----> 2 assert x%2==0, "El numero debe ser par"

AssertionError: El numero debe ser par
```

Manejo de Archivos

In [114…
```
#Para abrir un archivo utilizamos la funcion open
f = open("prueba.txt")
#Para poder leer su contenido utilizamos la funcion read(), tambien podemos especificar cu
f.read()
```

Out[114…
```
'Hola Mundo\n'
```

In [116…
```
#Una vez que querramos cerrar el archivo usamos la funcion close()
f.close()
```

In [122…

```
#Tambien podemos utilizar readline si solo quisieramos leer una solo linea
f = open("prueba.txt")
print(f.readline())
f.close()
```

```
Hola Mundo
```

In [126…

```
#Si quisieramos escribir en el archivo que hemos abierto lo podemos hacer de la siguiente
f = open("prueba.txt", "a")
f.write("Escribiendo nuevo contenido")
f.close()

f = open("prueba.txt", "r")
print(f.read())
f.close()
```

```
Hola Mundo
Escribiendo nuevo contenidoEscribiendo nuevo contenidoEscribiendo nuevo contenido
```

In [127…

```
#Tambien podemos sobreescribir todo el archivo
f = open("prueba.txt", "w")
f.write("Escribiendo nuevo contenido")
f.close()

f = open("prueba.txt", "r")
print(f.read())
f.close()
```

```
Escribiendo nuevo contenido
```

In [128…

```
#Tambien podemos crear un nuevo archivo
f = open("prueba2.txt","x")
f.close()
f = open("prueba2.txt", "w")
f.write("Escribiendo nuevo contenido en prueba2.txt")
f.close()

f = open("prueba2.txt", "r")
print(f.read())
f.close()
```

```
Escribiendo nuevo contenido en prueba2.txt
```

Manejo de Libreria OS

In [136…

```
#Para hacer uso de la libreria debemos importar el modulo
import os

#Obtener el directorio actual de trabajo
cwd= os.getcwd()
print("Directorio actual ", cwd)

# Para cambiar el directorio
os.chdir('../')
cwd= os.getcwd()
print("Cambio de directorio a ", cwd)
```

```
Directorio actual  D:\Development\Coding\Python\BigData
Cambio de directorio a  D:\Development\Coding\Python
```

In [131…
```
os.chdir("BigData")

#Para crear un nuevo folder
os.mkdir("DirectorioPrueba")

#Para listar todos los archivos y directorios
print(os.listdir(os.getcwd()))
```

```
['.ipynb_checkpoints', 'DirectorioPrueba', 'prueba.txt', 'prueba2.txt', 'RepasoPython.ipynb', 'RepasoPython.pdf']
```

In [132…
```
#Si queremos remover un archivo usamos
os.remove('prueba2.txt')
print(os.listdir(os.getcwd()))
```

```
['.ipynb_checkpoints', 'DirectorioPrueba', 'prueba.txt', 'RepasoPython.ipynb', 'RepasoPython.pdf']
```

In [133…
```
#Para eliminar un folder
os.rmdir('DirectorioPrueba')
print(os.listdir(os.getcwd()))
```

```
['.ipynb_checkpoints', 'prueba.txt', 'RepasoPython.ipynb', 'RepasoPython.pdf']
```

In [134…
```
os.name
```

Out[134…
```
'nt'
```

In [135…
```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print(arr)

print(type(arr))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

Uso de la Libreria Pandas

In [140…
```
#Importamos la libreria pandas y le asignamos un alias
import pandas as pd

#Dataset Kaggle
dataset = {
  'car_name': ["Hyundai i20", "Mahindra Scorpio Classic", "Citroen C3"],
  'fuel_type': ["Petrol", "Diesel", "Petrol"]
}

#Creacion de un dataframe
x = pd.DataFrame(dataset)
x
```

Out[140…

	car_name	fuel_type
0	Hyundai i20	Petrol
1	Mahindra Scorpio Classic	Diesel

	car_name	fuel_type
2	Citroen C3	Petrol

In [142…
```
#Para ver la version de pandas
print(pd.__version__)
```

```
1.3.4
```

Series

In [148…
```
#Generar una serie, la cual es como una columna en una tabla
a = [1,2,3]
x=pd.Series(a)
x
```

Out[148…
```
0    1
1    2
2    3
dtype: int64
```

In [150…
```
#Podemos crear etiquetas
x=pd.Series(a,index=["x","y","z"])
x
```

Out[150…
```
x    1
y    2
z    3
dtype: int64
```

In [152…
```
x["y"]
```

Out[152…
```
2
```

In [153…
```
x[1]
```

Out[153…
```
2
```

In [154…
```
# Tambien podemos crear una serie de un diccionario
temperaturas = {"dia1":30,"dia2":28,"dia3":32}
x=pd.Series(temperaturas)
x
```

Out[154…
```
dia1    30
dia2    28
dia3    32
dtype: int64
```

In [156…
```
# Tambien podemos crear una serie solo usando ciertos elementos del diccionario
temperaturas = {"dia1":30,"dia2":28,"dia3":32}
x=pd.Series(temperaturas, index=["dia1","dia2"])
x
```

Out[156…
```
dia1    30
dia2    28
dtype: int64
```

Dataframes

In [158…

```
dataset = {
  'car_name': ["Hyundai i20", "Mahindra Scorpio Classic", "Citroen C3"],
  'fuel_type': ["Petrol", "Diesel", "Petrol"]
}

#Creacion de un dataframe
x = pd.DataFrame(dataset)
x
```

Out[158…

	car_name	fuel_type
0	Hyundai i20	Petrol
1	Mahindra Scorpio Classic	Diesel
2	Citroen C3	Petrol

In [160…

```
#Podemos obtener una fila epecifica de la siguiente manera
x.loc[0]
```

Out[160…

```
car_name      Hyundai i20
fuel_type          Petrol
Name: 0, dtype: object
```

In [163…

```
#Podemos obtener multiples fila especificas
x.loc[[0,1]]
```

Out[163…

	car_name	fuel_type
0	Hyundai i20	Petrol
1	Mahindra Scorpio Classic	Diesel

Lectura CSV

In [165…

```
os.chdir("BigData")
#Para leer un archivo csv
df = pd.read_csv('Methane_final.csv')
df
```

Out[165…

	Unnamed: 0	region	country	emissions	type	segment	reason	baseYear	notes
0	0	Africa	Algeria	257.611206	Agriculture	Total	All	2019-2021	Average based on United Nations Framework Conv...
1	1	Africa	Algeria	0.052000	Energy	Bioenergy	All	2022	Estimates from end-uses are for 2020 or 2021 (...
2	2	Africa	Algeria	130.798996	Energy	Gas pipelines and LNG facilities	Fugitive	2022	Not available

	Unnamed: 0	region	country	emissions	type	segment	reason	baseYear	notes
3	3	Africa	Algeria	69.741898	Energy	Gas pipelines and LNG facilities	Vented	2022	Not available
4	4	Africa	Algeria	213.987000	Energy	Onshore gas	Fugitive	2022	Not available
...	...	...	...	...	...	...	...	...	...
1543	1543	World	World	3102.500000	Energy	Satellite-detected large oil and gas emissions	All	2022	Not available
1544	1544	World	World	30296.500000	Energy	Steam coal	All	2022	Not available
1545	1545	World	World	133350.984375	Energy	Total	All	2022	Estimates from end-uses are for 2020 or 2021 (...
1546	1546	World	World	9737.874023	Other	Total	All	2019-2021	Average based on United Nations Framework Conv...
1547	1547	World	World	70758.710938	Waste	Total	All	2019-2021	Average based on United Nations Framework Conv...

1548 rows × 9 columns

Lectura JSON

In [196…
```
import json
f = open("prueba.json")
x = json.load(f)
f.close()
x[0]
```

Out[196…
```
{'questionType': 'yes/no',
 'asin': '1466736038',
 'answerTime': 'Mar 8, 2014',
 'unixTime': 1394265600,
 'question': 'Is there a SIM card in it?',
 'answerType': 'Y',
 'answer': 'Yes. The Galaxy SIII accommodates a micro SIM card.'}
```

In [197…
```
x[0]['questionType']
```

Out[197…
```
'yes/no'
```

In [200…
```
#Para leer un json
df = pd.read_json('prueba.json')
df.head(2)
```

Out[200…

	questionType	asin	answerTime	unixTime	question	answerType	answer
0	yes/no	1466736038	Mar 8, 2014	1394265600	Is there a SIM card in it?	Y	Yes. The Galaxy SIII accommodates a micro SIM ...
1	open-ended	1466736038	Aug 4, 2014	1407135600	Why hasnt it upgraded to latest Android OS 4.4...	NaN	My S3 was able to upgrade to 4.4.2 last week, ...

Analisis de Data

In [202…

```
#Para ver cierta cantidad de datos desde el inicio usamos la funcion head
df.head(3)
```

Out[202…

	questionType	asin	answerTime	unixTime	question	answerType	answer
0	yes/no	1466736038	Mar 8, 2014	1394265600	Is there a SIM card in it?	Y	Yes. The Galaxy SIII accommodates a micro SIM ...
1	open-ended	1466736038	Aug 4, 2014	1407135600	Why hasnt it upgraded to latest Android OS 4.4...	NaN	My S3 was able to upgrade to 4.4.2 last week, ...
2	yes/no	1466736038	Jan 29, 2015	1422518400	Is this phone new, with 1 year manufacture war...	?	It is new but I was not able to get it activat...

In [204…

```
#Para ver cierta cantidad de datos desde el final usamos la funcion tail
df.tail(3)
```

Out[204…

	questionType	asin	answerTime	unixTime	question	answerType	answer
9	yes/no	1466736038	Sep 3, 2014	1409727600	will this phone work with straight talk?	?	As an unlocked phone it is capable of working ...
10	yes/no	1621911888	Dec 13, 2013	1386921600	Is it unlocked?	Y	yes
11	yes/no	1621911888	Dec 13, 2013	1386921600	Is it international?	Y	yes it is international.

In [205…

```
#Para tener información de la data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12 entries, 0 to 11
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   questionType  12 non-null     object
 1   asin          12 non-null     int64 
 2   answerTime    12 non-null     object
 3   unixTime      12 non-null     int64 
 4   question      12 non-null     object
 5   answerType    8 non-null      object
 6   answer        12 non-null     object
```

```
dtypes: int64(2), object(5)
memory usage: 800.0+ bytes
```

Para limpieza de datos

In [206…

```
#Obtener el dataframe sin resultados vacios usamos la funcion dropna, si quisieraramos car
# usamos inplace=True
new_df=df.dropna()
new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 8 entries, 0 to 11
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   questionType  8 non-null      object
 1   asin          8 non-null      int64 
 2   answerTime    8 non-null      object
 3   unixTime      8 non-null      int64 
 4   question      8 non-null      object
 5   answerType    8 non-null      object
 6   answer        8 non-null      object
dtypes: int64(2), object(5)
memory usage: 512.0+ bytes
```

In [207…

```
new_df
```

Out[207…

	questionType	asin	answerTime	unixTime	question	answerType	answer
0	yes/no	1466736038	Mar 8, 2014	1394265600	Is there a SIM card in it?	Y	Yes. The Galaxy SIII accommodates a micro SIM ...
2	yes/no	1466736038	Jan 29, 2015	1422518400	Is this phone new, with 1 year manufacture war...	?	It is new but I was not able to get it activat...
3	yes/no	1466736038	Nov 30, 2014	1417334400	can in it be used abroad with a different carr...	Y	Yes
4	yes/no	1466736038	Nov 24, 2014	1416816000	Is this phone brand new and NOT a mini?	?	The phone we received was exactly as described...
7	yes/no	1466736038	Oct 2, 2014	1412233200	Does this phone use the regular Sim card (the ...	?	it takes mini sim
9	yes/no	1466736038	Sep 3, 2014	1409727600	will this phone work with straight talk?	?	As an unlocked phone it is capable of working ...
10	yes/no	1621911888	Dec 13, 2013	1386921600	Is it unlocked?	Y	yes
11	yes/no	1621911888	Dec 13, 2013	1386921600	Is it international?	Y	yes it is international.

In [209…

```
#Si en lugar de eliminar esa filas que contengan vacios, queremos reemplazar con ciertos v
df_new= df
df_new["answerType"].fillna('N',inplace=True)
df_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 12 entries, 0 to 11
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   questionType  12 non-null     object
 1   asin          12 non-null     int64 
 2   answerTime    12 non-null     object
 3   unixTime      12 non-null     int64 
 4   question      12 non-null     object
 5   answerType    12 non-null     object
 6   answer        12 non-null     object
dtypes: int64(2), object(5)
memory usage: 800.0+ bytes
```

In [210…

```
df_new
```

Out[210…

	questionType	asin	answerTime	unixTime	question	answerType	answer
0	yes/no	1466736038	Mar 8, 2014	1394265600	Is there a SIM card in it?	Y	Yes. The Galaxy SIII accommodates a micro SIM ...
1	open-ended	1466736038	Aug 4, 2014	1407135600	Why hasnt it upgraded to latest Android OS 4.4...	N	My S3 was able to upgrade to 4.4.2 last week, ...
2	yes/no	1466736038	Jan 29, 2015	1422518400	Is this phone new, with 1 year manufacture war...	?	It is new but I was not able to get it activat...
3	yes/no	1466736038	Nov 30, 2014	1417334400	can in it be used abroad with a different carr...	Y	Yes
4	yes/no	1466736038	Nov 24, 2014	1416816000	Is this phone brand new and NOT a mini?	?	The phone we received was exactly as described...
5	open-ended	1466736038	Nov 3, 2014	1415001600	What is the warranty on this?	N	No warranty
6	open-ended	1466736038	Oct 14, 2014	1413270000	this product is used with GSM chip in my count...	N	I am sure (but not positive) that this phone w...
7	yes/no	1466736038	Oct 2, 2014	1412233200	Does this phone use the regular Sim card (the ...	?	it takes mini sim
8	open-ended	1466736038	Sep 11, 2014	1410418800	how much time you need to send me this product...	N	If you choose expedited shipping you will have...
9	yes/no	1466736038	Sep 3, 2014	1409727600	will this phone work with straight talk?	?	As an unlocked phone it is capable of working ...
10	yes/no	1621911888	Dec 13, 2013	1386921600	Is it unlocked?	Y	yes
11	yes/no	1621911888	Dec 13, 2013	1386921600	Is it international?	Y	yes it is international.

In [214…

```
#Para filtrar la filas que cumplan la condicion
```

```
df_new[df_new["answerType"]=='N']
```

Out[214…

	questionType	asin	answerTime	unixTime	question	answerType	answer
1	open-ended	1466736038	Aug 4, 2014	1407135600	Why hasnt it upgraded to latest Android OS 4.4...	N	My S3 was able to upgrade to 4.4.2 last week, ...
5	open-ended	1466736038	Nov 3, 2014	1415001600	What is the warranty on this?	N	No warranty
6	open-ended	1466736038	Oct 14, 2014	1413270000	this product is used with GSM chip in my count...	N	I am sure (but not positive) that this phone w...
8	open-ended	1466736038	Sep 11, 2014	1410418800	how much time you need to send me this product...	N	If you choose expedited shipping you will have...

In [221…

```
#Para filtrar por nombre de columna
df_new.filter(items=['answerType','question'])
```

Out[221…

	answerType	question
0	Y	Is there a SIM card in it?
1	N	Why hasnt it upgraded to latest Android OS 4.4...
2	?	Is this phone new, with 1 year manufacture war...
3	Y	can in it be used abroad with a different carr...
4	?	Is this phone brand new and NOT a mini?
5	N	What is the warranty on this?
6	N	this product is used with GSM chip in my count...
7	?	Does this phone use the regular Sim card (the ...
8	N	how much time you need to send me this product...
9	?	will this phone work with straight talk?
10	Y	Is it unlocked?
11	Y	Is it international?

In [220…

```
#Para seleccionar columnas
df_new[['answerType','question']]
```

Out[220…

	answerType	question
0	Y	Is there a SIM card in it?
1	N	Why hasnt it upgraded to latest Android OS 4.4...
2	?	Is this phone new, with 1 year manufacture war...
3	Y	can in it be used abroad with a different carr...
4	?	Is this phone brand new and NOT a mini?
5	N	What is the warranty on this?

	answerType	question
6	N	this product is used with GSM chip in my count...
7	?	Does this phone use the regular Sim card (the ...
8	N	how much time you need to send me this product...
9	?	will this phone work with straight talk?
10	Y	Is it unlocked?
11	Y	Is it international?

In [222...

```
#Para reemplazar un valor especifico
df_new.loc[1, 'answerType'] = 'Y'
df_new.head(3)
```

Out[222...

	questionType	asin	answerTime	unixTime	question	answerType	answer
0	yes/no	1466736038	Mar 8, 2014	1394265600	Is there a SIM card in it?	Y	Yes. The Galaxy SIII accommodates a micro SIM ...
1	open-ended	1466736038	Aug 4, 2014	1407135600	Why hasnt it upgraded to latest Android OS 4.4...	Y	My S3 was able to upgrade to 4.4.2 last week, ...
2	yes/no	1466736038	Jan 29, 2015	1422518400	Is this phone new, with 1 year manufacture war...	?	It is new but I was not able to get it activat...